A SIMULATION OPTIMIZATION APPROACH TO THE DESIGN OF UNMANNED AERIAL VEHICLES

THESIS

Emily C. Evans

AFIT/GOR/ENS/08-22

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

AFIT/GOR/ENS/08-22

A SIMULATION OPTIMIZATION APPROACH TO THE DESIGN OF UNMANNED AERIAL VEHICLES

THESIS

Presented to the Faculty

Department of Operational Sciences

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Education and Training Command

In Partial Fulfillment of the Requirements for the

Degree of Master of Science in Operations Research

Emily C. Evans, BISE

March 2008

AFIT/GOR/ENS/08-22

A SIMULATION OPTIMIZATION APPROACH TO THE DESIGN OF UNMANNED AERIAL VEHICLES

Emily C. Evans, BISE

Approved:

Dr. Sharif Melouk (Advisor)	date
Shane Hall, Maj, USAF (Reader)	date

AFIT/GOR/ENS/08-22

Abstract

Military strategy and operations have evolved significantly over the past decade. This evolution has led to a change in the military resources required to carry out missions successfully. In line with these requirements, demand has increased for unmanned aerial vehicles (UAV) with enhanced capability to perform surveillance and to strike targets of interest. This research effort aids in the design of a next generation UAV by employing a simulation optimization approach. The goal of this research is to maximize the number of targets destroyed in a conflict scenario by a newly designed UAV that is subject to size, weight, and budget constraints. The solution approach involves the development of a simulation model representing a conflict scenario, which includes various types and quantities of targets, and weather conditions. The model is used to test the effectiveness of various UAV configurations in detecting and destroying targets. A tabu search meta-heuristic is constructed to optimize the configuration of the UAV, in terms of the number and type of sensors, synthetic aperture radar, and weapons.

AFIT/GOR/ENS/08-22

To my husband and parents

Acknowledgments

I would like to thank the members of my committee for their guidance and assistance in this thesis effort. Additionally, I would like to thank the Air Force Research Laboratory Air Vehicles Directorate for sponsoring this research effort.

Emily C. Evans

Table of Contents

Page

List of Figures

List of Tables

A SIMULATION OPTIMIZATION APPROACH TO THE DESIGN OF UNMANNED AERIAL VEHICLES

I. Introduction

1.1 Background

Simulation is a tool used to model and study complex systems. Simulation can be performed prior to the development of the actual system or without changing the existing system. A simulation receives a set of input values from the analyst, runs for a specified amount of time or number of replications, and outputs a performance measure.

The goal of optimization is to determine the values for a given set of decision variables that maximizes or minimizes an objective function subject to constraints. When optimization is applied to simulation, the resulting methodology is called simulation optimization. Simulation optimization attempts to determine the simulation input values that maximizes or minimizes the simulation performance measure subject to problem constraints.

In the past, a “Catch 22” existed in respect to simulation optimization (Glover et al. 1999). Problems requiring simulation modeling were extremely complex. The complexity of these models hindered the application of traditional optimization methods. However, newer optimization methods are now being applied to simulation successfully. Today, simulation optimization is applied in many different arenas including:

manufacturing, workforce planning, facility location and design, and financial planning (Glover et al. 1999). For example, the use of simulation optimization in financial planning would allow analysts to determine what investments to make without risking the loss of an investor's money.

1.2 Research Motivation

Simulation optimization can also be applied to resource allocation problems. This application requires the simulation to describe a system in which the resources are the input parameters. The resources are allocated to processes according to the output of the optimizer. The simulation runs for a specified amount of time or number of replications and outputs a performance measure. Now, consider a problem in which the resources are modularized sensor and weapons packages. These resources are being allocated to unmanned aerial vehicles (UAVs), specifically the Revolutionary Hunter/Killer (Rev/HK) UAV.

The Rev/HK is a UAV concept that will perform both surveillance and strikes on targets of interest. It is anticipated that the Rev/HK will provide an unmanned aircraft that is persistent, i.e. the aircraft can remain in flight for extended periods of time, survivable, and responsive to targets of interest. Also, the Rev/HK design will incorporate technological advances in aircraft speed, stealth, and sensing capability (Morris 2006).

Technological advances in sensing capability will be considered in this research. The following sensors may be included on the Rev/HK: electro-optical (EO) sensors, infrared (IR) sensors, and synthetic aperture radar (SAR). EO sensors can function during the day and IR sensors can function during the day and night. Both EO and IR sensors, which will be simply referred to as sensors, are limited in sensing capability during inclement weather (Chaput 2002). SAR provides sensing capabilities during the day, night, and all weather conditions (Mileshosky 2005).

The sensors and weapons packages for the Rev/HK are modularized, allowing for different sensors and weapons to be used depending upon the simulated scenario. Because sensor packages represent "one of the single largest cost items in an unmanned aircraft," sensors are assigned to a UAV not only based upon the probability of mission success but also within a specific budgetary constraint ("Unmanned Aircraft Systems Roadmap" 2005). Additional constraints include a limitation on sensor weight and size. The weapons package distribution and the number of UAVs used in the simulation scenario will also be taken into account in this research.

1.3 Research Objectives

This research aims to develop and analyze the results of a simulation optimization framework for a resource allocation problem, specifically the Rev/HK problem. This research will address the following objectives:

1. Develop a methodology that incorporates both simulation and optimization into a resource allocation problem.
2. Test the robustness of the methodology using design of experiments (DOE).
3. Determine the optimal or near optimal input parameters given specific simulation scenarios.

1.4 Research Approach

A simulation optimization approach will be used to ascertain the best sensor package, weapons distribution, and number of UAVs for a simulated scenario. Simulation optimization is an iterative process; the simulation describes a scenario in which the initial input parameters are a feasible combination of sensor and weapon components for a specified number of Rev/HK UAVs. Given the mission goal of detecting and destroying targets, the simulation outputs the percentage of targets destroyed and the time required to achieve a solution. An optimizer, in this case a tabu search metaheuristic, then changes the combination of sensors, weapons, and number of UAVs in an attempt to maximize the percentage of targets destroyed subject to size, weight, and budget constraints. The tabu search metaheuristic provides the number of UAVs as well as a new set of sensor and weapon components as input parameters to the simulation. The iterative process continues until the simulation is either run for a maximum amount of time, maximum number of iterations, or no longer achieves an improved solution within a specified number of iterations.

DOE will provide a method to test the robustness of the algorithm. In the DOE, the tabu search parameters are factors, the different simulation scenarios are blocks, and the percentage of targets destroyed and the time required to achieve a solution are the responses. The goal is to create an algorithm that is effective for various sizes of simulation scenarios. To determine the best input parameter settings, the simulation optimization procedure is run at the robust tabu search parameter settings.

1.5 Research Impact

This research will provide a simulation optimization methodology for resource allocation problems, specifically, the Rev/HK problem. The algorithms and computer code developed for the Rev/HK problem will serve as UAV development and design tools in the future. Additionally, this solution approach can serve as an example to other researchers or analysts addressing similar problems. The testing procedure will demonstrate how to develop a robust simulation optimization framework.

1.6 Organization

Chapter II provides an overview of the current literature relating to simulation optimization. Chapter III outlines the specific simulation optimization methodology used in this research. Chapter IV presents the results obtained from the implementation of the methodology discussed in Chapter III. Chapter V provides concluding remarks and suggestions for future research. Appendix A presents the tabu search MATLAB code used in this research effort.

II. Literature Review

2.1 Introduction

Simulation optimization provides a means to determine the simulation input parameters that achieve optimal or near optimal simulation output. This chapter describes current literature regarding simulation optimization. Simulation optimization can employ a variety of optimizers that cater to the specifics of the problem being solved. Therefore, this chapter discusses optimization methods used in simulation optimization. This research will employ a tabu search metaheuristic as the optimization method; therefore, the specifics of the tabu search method will be described. In order to use the tabu search metaheuristic in this research, different aspects of other optimizers are incorporated into the metaheuristic. These aspects will be discussed in detail. Finally, simulation optimization testing procedures will be explored.

2.2 Simulation Optimization

According to Banks (1998) simulation is

> the imitation of the operation of a real-world process or system over time. Simulation involves the generation of an artificial history of the system and the observation of that artificial history to draw inferences concerning the operating characteristics of the real system that is represented.

A simulation model can incorporate the inherent uncertainties associated with real-world processes. By including natural uncertainties, very complex models can be created and studied (Glover et al. 1999).

One question often encountered within the study of simulation is how does one determine the best settings for a simulation, i.e., what given set of inputs provides the desired or best simulation output? This question leads to the area of study known as simulation optimization. Fu (2001) defines simulation optimization as the "optimization of performance measures based upon the outputs of stochastic…simulations." Fu (2002) further characterizes the optimization portion of simulation optimization as "optimization." The purpose of the quotation marks is to emphasize that the goal of the optimization procedure is provide new inputs that improve the outputs of the simulation; however, because of the stochastic nature of the simulation, there is "no way of knowing if an optimal has actually been reached" (Fu 2002).

Fu et al. (2005) describe simulation optimization in terms of a general optimization model. The goal of the general optimization model is "to find a setting of controllable parameters that minimizes a given objective function" (Fu et al. 2005). When the optimization problem is applied to simulation optimization, the goal remains the same; however, the objective function value is now estimated by the simulation model.

Simulation optimization is an iterative process. The simulation model describes the system that is being studied. Additionally, the simulation model provides outputs that are used to create the objective function that is evaluated by the optimizer (Fu 2002). The optimizer changes the simulation inputs with respect to the problem constraints in order to improve the output of the simulation (Fu 2001). The process is continued for either a specified amount of time, a set number of iterations, or until an acceptable objective function value is reached.

Simulation optimization can prove to be computationally expensive. Because of the iterative nature of the process, many objective function evaluations are required.

Each objective function evaluation requires the simulation to be run for a specified number of replications. Therefore, a tradeoff is required between the "amount of computational effort needed to estimate the performance at a particular solution versus the effort in finding improved solution points" (Fu 2002).

2.3 Optimization Methods

The optimization techniques applied to simulation optimization problems have two requirements: (1) the technique should provide convergence to an optimal solution even though there may be significant noise in the estimation of the desired performance measures and (2) the method should provide convergence in a reasonable amount of time (Andradóttir 1998). The optimization methods can be divided into three categories: statistical procedures, stochastic optimization, and metaheuristics (Fu 2001).

2.3.1 Statistical Procedures

Ranking and selection (R&S) and sequential response surface methodology (RSM) are statistical optimization techniques that can be used in the simulation optimization framework. R&S is applicable when considering a fixed set of configuration alternatives (Fu et al. 2005) where the number of configurations is relatively small, between two and 20 configurations (Goldsman and Nelson 1998). R&S can also be used in conjunction with other simulation optimization methods. R&S can be used as a screening method to reduce the number of configurations prior to simulation optimization (Fu et al. 2005). Additionally, Boesel et al. (2003) propose a method in which R&S is used to "clean up" after simulation optimization. This proposed method determines the best configuration after a search based optimization method has been used.

According to Kleijnen (1998), in order to perform simulation optimization using RSM, the sequential RSM technique must be used. Sequential RSM creates a localized response surface and corresponding metamodel (April et al. 2003) using regression or neural networks (Fu 2002). A search strategy is determined using the metamodel until a "linear fit is deemed adequate" (Fu 2001). Next, additional points are simulated in order to estimate the optimum (Fu 2001).

2.3.2 Stochastic Optimization

Stochastic approximation and random search are stochastic optimization techniques. Stochastic approximation, typically applied to continuous variable problems, is similar to the gradient search methods used in discrete optimization (April et al. 2003). However, because stochastic approximation mimics the gradient search method, it often finds local optima rather than the global optimum (Fu 2002). Random search is an iterative process where the inputs change from the current point to somewhere in the neighborhood of that point (April et al. 2003). Random search algorithms differ in respect to how the neighborhood is defined, how the next point is chosen, and how the optimal is estimated (Fu 2005).

2.3.3 Metaheuristics

Metaheuristics commonly used in conjunction with simulation optimization include: genetic algorithms, simulated annealing, and tabu search. This research will focus on the tabu search metaheuristic; therefore, tabu search will be discussed in detail in Section 2.4. Genetic algorithms, introduced by Holland in 1975 (Mühlenbein 2003), are “inspired by Darwinian theory” (Sait and Youssef 1999a). The algorithm “emulates the natural process of evolution to perform … [a] systematic search … toward the optimum” (Sait and Youssef 1999a).

The genetic algorithm begins with a set of solutions, known as the population. Within an iteration, two parent solutions recombine, or crossover, to create offspring. The offspring form the next generation of solutions (Banks et al. 2005). The new solutions will hopefully provide better objective function values than those of the previous generations, resulting in the survival of the fittest.

Simulated annealing is a metaheuristic based upon the physical annealing process. The physical annealing process is a “thermal process for obtaining low-energy states of a solid in a heat bath” (Aarts et al. 2003). Kirkpatrick, Gelatt, and Vecchi applied the simulation of the physical process to optimization in 1983, resulting in the simulated annealing metaheuristic (Eglese 1990).

The simulated annealing metaheuristic uses much of the same terminology as the physical annealing process. The parameter, temperature, slowly decreases throughout the algorithm in an attempt to achieve a global minimum. A cooling schedule determines how fast the parameter temperature decreases. Additionally, the objective function that is evaluated describes the state of the system (Anandalingam 2001). The simulated annealing metaheuristic accepts increases in the objective function value on a “probabilistic basis” in order to allow for different areas of the solution space to be evaluated (Sait and Youssef 1999c). The consideration of different parts of the solution space occurs with the hope that the algorithm will drive the solution to the global minimum (Anandalingam 2001).

2.4 Tabu Search

The tabu search metaheuristic, introduced by Glover in 1986, incorporates adaptive memory into the algorithm (Hertz et. al 2003). Because adaptive memory is employed in the algorithm, one can clearly see how the origins of the tabu search metaheuristic lie in the logic of artificial intelligence and human thought (Sait and Youssef 1999d).

2.4.1 Tabu Search Terminology

In order to develop new solutions, the tabu search algorithm performs moves in specified neighborhoods. Given a current solution, a move is an operation performed on the current solution to create a new solution (Glover and Laguna 1997). Neighborhoods contain all of the potential moves that can be performed on the given solution (Sait and Youssef 1999b).

In order to determine the next move, the tabu search algorithm searches through a neighborhood to determine quality moves (Glover and Laguna 1997). After being considered by the algorithm, previous moves are stored in the tabu list. While a move is considered tabu, the algorithm cannot perform the move again. Moves located on the tabu list are considered tabu for a specified amount of time, which is known as the tabu tenure. A tabu move is allowed when a specific condition, the aspiration criterion, is achieved (Sait and Youssef 1999d). Often, the aspiration criterion dictates that a tabu move can be used if the resulting objective function value is the best thus far.

Additional features of the tabu search algorithm include diversification and intensification. Diversification allows the algorithm to explore different regions of the solution space (Glover 2001). An elite candidate list holds the solutions that have produced the best objective function value thus far. Members of the elite candidate list are often used in the intensification process, allowing the algorithm to search around solutions that have previously resulted in quality objective function values (Hertz et. al 2003).

2.4.2 Tabu Search Hybrid

This research uses a hybrid tabu search procedure as the optimizer in the simulation optimization methodology. As previously stated, a pure tabu search metaheuristic searches through a neighborhood to find a quality move. However, the metaheuristic used in this research uses random draws to determine the next move. This is done because of the amount of computational effort required to determine the objective function of one move. Note that in order to calculate a single objective function value, the simulation runs for a number of replications. The random choice of moves within a neighborhood creates a tabu search hybrid utilizing the random draw aspects of random search and simulated annealing (Glover and Laguna 1997).

2.5 Testing Procedures

According to Greenberg (1990), computational testing should be performed in order to ensure the "correctness of the model or algorithm, the quality of the solution, the speed of computation, and the robustness" of the model. Additionally, Greenberg (1990) states that statistical analysis, through the use of design of experiments (DOE), can validate and verify a simulation model.

Hooker (1995) states that the current focus in developing metaheuristics is based on competition to see who can develop a newer, faster algorithm. Hooker (1995) likens this competitive current testing procedure to a track race and deems it "anti-intellectual." Instead of the track race approach, he suggests a statistical approach using DOE for testing heuristics. Because DOE is a recommended procedure for testing both simulation and optimization it should work well in the simulation optimization framework. In order to test the simulation optimization framework developed in this research, a full factorial DOE with blocking is utilized.

2.6 Conclusion

This chapter describes simulation optimization as well as different optimization methods used in the simulation optimization framework. Additionally, the key aspects of tabu search are explained and an adapted tabu search metaheuristic is described. Finally, testing procedures for simulation optimization are explained.

III. Methodology

3.1 Introduction

Simulation models allow analysts and researchers to produce complex scenarios without changing the actual systems being modeled. A tabu search metaheuristic, using adaptive memory to develop new solutions, can be incorporated into a simulation framework: the result is an iterative simulation optimization methodology. A simulation optimization methodology attempts to find the optimal or near optimal solutions to very complex real world problems. The methodology presented in this chapter will apply the simulation optimization method to the Rev/HK problem.

First, this chapter presents an overview of the simulation optimization methodology. Next, the chapter provides a detailed explanation of the simulation that describes the flight and military actions associated with the Rev/HK UAVs. Additionally, an optimization formulation is described such that constraints are placed on the inputs to the simulation. A tabu search algorithm is provided and different key tabu search terms and parameters are defined in respect to the problem. Next, the testing procedure for the simulation optimization method is outlined. Finally, an example problem is presented to demonstrate how the methodology is implemented.

3.2 Simulation Optimization Overview

Simulation optimization provides a methodology to determine the input parameters to be used in a simulation in order to produce optimal or near optimal simulation outputs. The simulation optimization process is composed of two components: the stochastic simulation and the optimizer. First, the simulation describes the process being studied. The simulation requires input parameters to describe the scenario being evaluated. The input parameters are typically provided by the analyst using the simulation. The simulation is run for a specified amount of time or a set number of replications and outputs a performance measure.

The goal of the optimizer is to provide a set of input parameters such that the simulation outputs the best possible performance measure. The optimizer provides new sets of input parameters to the simulation based upon constraints defined by the problem. The optimizer views the output of the simulation as the objective function value. As previously stated, simulation optimization is an iterative process. The optimizer continues to provide new potential input parameter settings to the simulation until a defined stopping criterion is satisfied. Figure 3-1, from Law (2007), provides a description of the simulation optimization process.

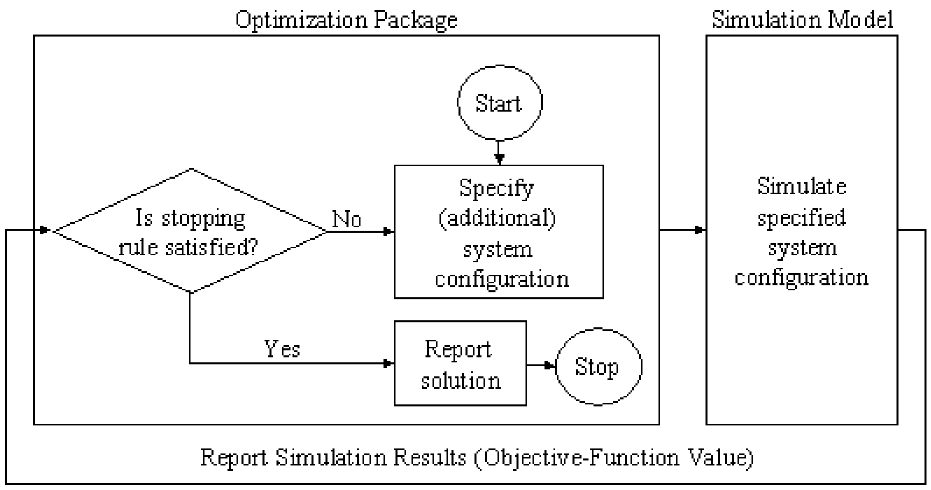
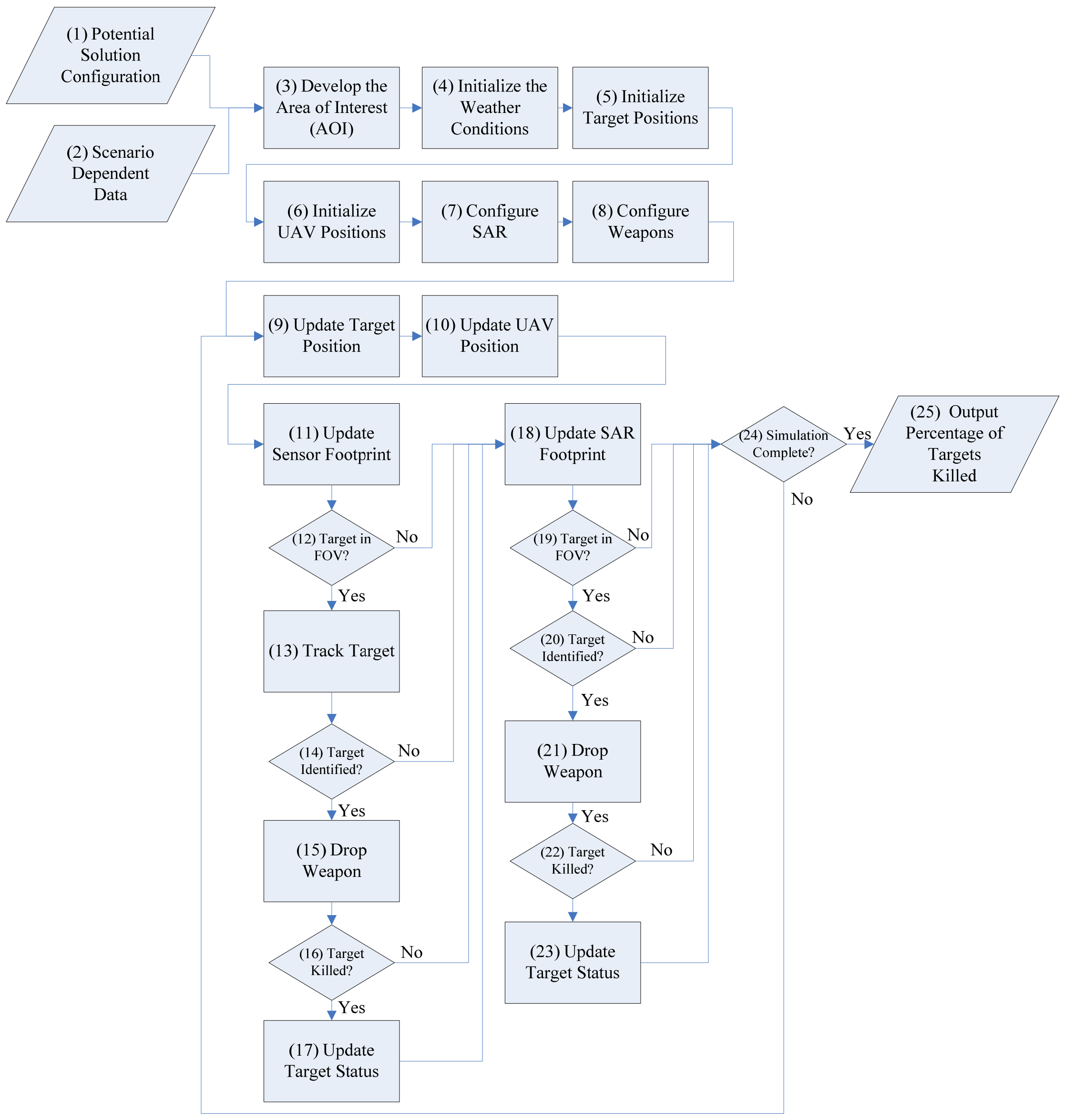

Optimization Package
Simulation Model
Start
Specify (additional) system configuration
Is stopping rule satisfied?
No
Yes
Report solution
Stop
Simulate specified system configuration
Report Simulation Results (Objective-Function Value)

Figure 3-1: Simulation optimization process

3.3 Simulation Component

The Rev/HK simulation, implemented in MATLAB, describes the combat scenario and the flight and military actions of the UAV. Figure 3-2 provides a flowchart of the simulation.



(1) Potential Solution Configuration
(2) Scenario Dependent Data
(3) Develop the Area of Interest (AOI)
(4) Initialize the Weather Conditions
(5) Initialize Target Positions
(6) Initialize UAV Positions
(7) Configure SAR
(8) Configure Weapons
(9) Update Target Position
(10) Update UAV Position
(11) Update Sensor Footprint
(12) Target in FOV?
No
Yes
(13) Track Target
(14) Target Identified?
No
Yes
(15) Drop Weapon
(16) Target Killed?
No
Yes
(17) Update Target Status
(18) Update SAR Footprint
(19) Target in FOV?
No
Yes
(20) Target Identified?
No
Yes
(21) Drop Weapon
Yes
(22) Target Killed?
No
(23) Update Target Status
(24) Simulation Complete?
Yes
No
(25) Output Percentage of Targets Killed

Figure 3-2: Simulation flowchart

3.3.1 Inputs

There are two separate inputs to the simulation: the configuration and scenario inputs. Figure 3-3 provides a detailed view of the input components in the simulation flowchart. The configuration inputs include the types of sensors and SAR included on the UAVs, the weapons distribution, and the number of UAVs. The configuration inputs are discussed further in Section 3.5 and are displayed as (1) in Figure 3-3. The scenario inputs describe the area of interest (AOI), target characteristics, weather conditions, flight pattern, maximum altitude, and maximum speed. The target characteristics are represented by a two element array. The first element is the target density, or the number of targets per square nautical mile. The second element is the percentage of targets that are vehicles. Weather conditions such as cloudy skies, fog, mist, haze, and rain can be modeled. The scenario inputs are displayed as (2) in Figure 3-3.

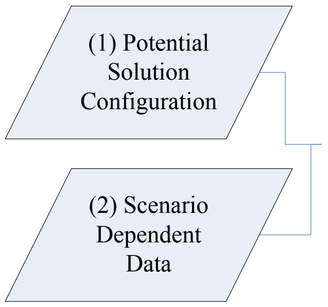

(1) Potential Solution Configuration
(2) Scenario Dependent Data

Figure 3-3: Input portion of the simulation flowchart

3.3.2 Initializations

After the simulation receives the necessary inputs, different aspects of the simulation require initialization (provided in Figure 3-4, a detailed view of the initialization portion of the simulation flowchart): the AOI (3), the weather conditions (4), the target positions (5), and the UAV potions (6). The initialization of the AOI develops the area over which the targets will be scattered and the UAVs will search. The desired weather conditions and targets are randomly placed according to a uniform distribution throughout the AOI. The initialization of the UAVs involves assigning their altitude, speed, search pattern, starting location, and sensor set. Each UAV is assigned an identical sensor set. Additionally, the UAVs are assigned identical SAR and weapons distributions that are configured in (7) and (8) of Figure 3-4, respectively.

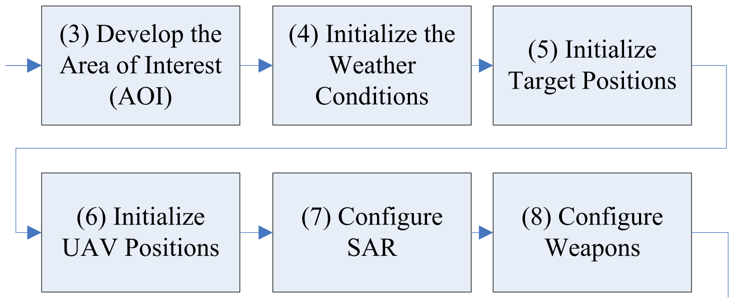

(3) Develop the Area of Interest (AOI)
(4) Initialize the Weather Conditions
(5) Initialize Target Positions
(6) Initialize UAV Positions
(7) Configure SAR
(8) Configure Weapons

Figure 3-4: Initialization portion of the simulation flowchart

3.3.3 Target and UAV Position Update

After all initializations and assignments are made, the target (9) and UAV positions (10) are updated in Figure 3-5 (a detailed view of the position update portion of the simulation flowchart). Target directions and movements are determined randomly. UAV movements are determined according to the given altitude, speed, and search pattern.

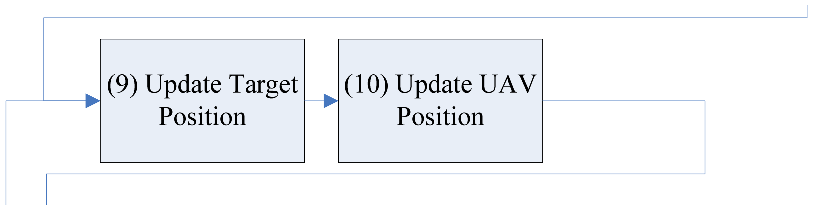


Figure 3-5: Update portion of the simulation flowchart

3.3.4 Sensor Update

Next, the use of the EO/IR sensors is considered. Figure 3-6 provides a detailed view of the sensor update portion of the simulation flowchart. First, the sensor footprints are updated (11 in Figure 3-6). The sensor footprints represent the area of the AOI that the sensors can view. Consider a simulation where one sensor is assigned to each UAV. The simulation will evaluate if a target is in the sensor's field of view (FOV) (12 in Figure 3-6). If weather does not impede the sensor's capabilities and a target is in the sensor's FOV, the sensor will track the target (13 in Figure 3-6); however, if a target is not in the sensor's FOV, the simulation will continue to the next step and update the SAR footprint (18 in Figure 3-6). If the target is tracked, the simulation will assign the target probabilities of detection, recognition, and identification based upon Johnson's Criteria (14 in Figure 3-6) (Chaput 2002). If the probability of identification is one, the UAV will drop a weapon on the target (15 in Figure 3-6); however, if the target is not identified, the simulation will move to the next step and update the SAR footprint (18 in Figure 3-6). If a weapon is dropped, the target may or may not be destroyed (16 in Figure 3-6). If the target is destroyed, the target status is updated to "killed" (17 in Figure 3-6) and the process continues to update the SAR footprint (18 in Figure 3-6). However, if the target is not destroyed, the simulation will move to the next step and update the SAR footprint (18 in Figure 3-6).

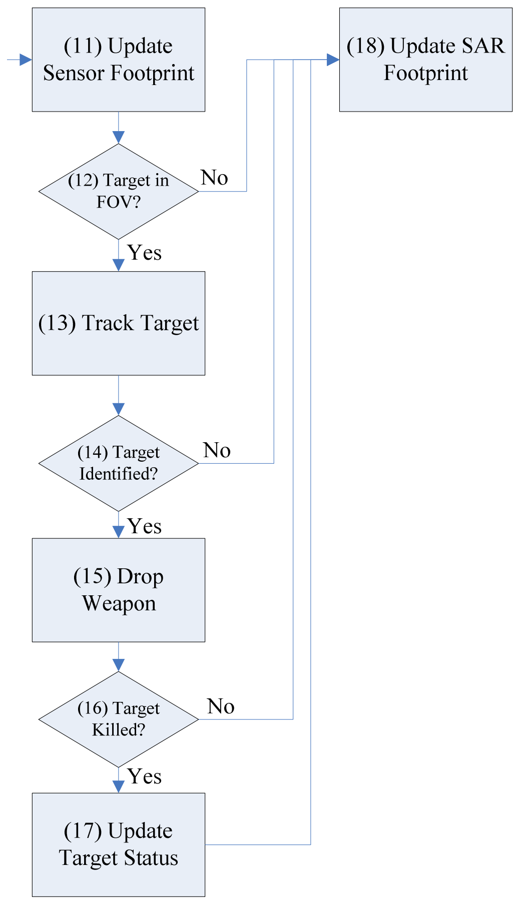


Figure 3-6: Sensor update portion of the simulation flowchart

3.3.5 SAR Update and Simulation Completion

The SAR is evaluated in a similar fashion as the EO/IR sensors. Figure 3-7 provides a detailed version of the SAR update and simulation completion from the simulation flowchart. First, the simulation determines if a target is in the SAR footprint, which is the area of the AOI that the SAR can view (19 in Figure 3-7). If a target is not in the footprint, the simulation continues to the next step to determine if the simulation is complete, meaning that all UAVs have completed their search patterns (24 in Figure 3-7). However, if a target is in the footprint, the target is assigned a probability of detection, recognition, or identification according to the National Image Interpretability Rating Scale (20 in Figure 3-7) (Chaput 2002). If the target is identified with a probability of one, a weapon is dropped on the target (21 in Figure 3-7); however, if the target is not identified, the simulation continues to the next step to determine if the simulation is complete (24 in Figure 3-7). The SAR process now follows the exact same pattern as the sensor process. The target may or may not be destroyed (22 in Figure 3-7). If the target is destroyed, its status is updated to "killed" (23 in Figure 3-7). The process next moves to the determination of whether or not the simulation is complete (24 in Figure 3-7). If the simulation is not complete, the simulation will update the target positions (described in section 3.3.3) and the process will begin again.

The simulation is run until all UAVs have completed their assigned search patterns or for a maximum number of iterations. The simulation will output the percentage of targets destroyed (25 in Figure 3-7). The output will be used as the objective function value in the optimization portion of the problem.

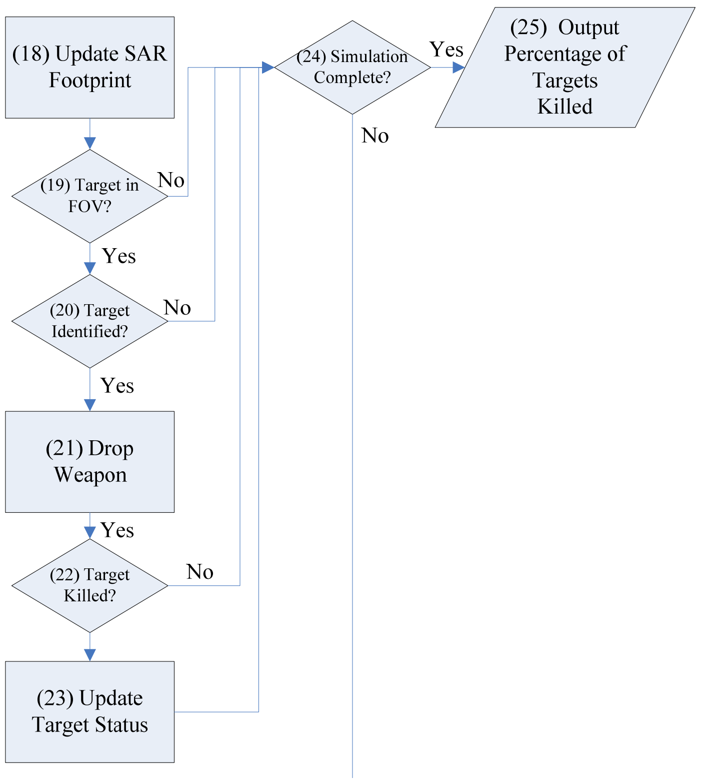

(18) Update SAR Footprint
(19) Target in FOV?
No
Yes
(20) Target Identified?
No
Yes
(21) Drop Weapon
Yes
(22) Target Killed?
No
(23) Update Target Status
(24) Simulation Complete?
Yes
No
(25) Output Percentage of Targets Killed

Figure 3-7: SAR update portion of the simulation flowchart

3.4 Optimization Component

The optimization problem associated with the previously described simulation is a mixed integer nonlinear program (MINLP). The MINLP includes an objective function that maximizes the percentage of targets destroyed. The objective function is subject to nine constraints, which include: budget, size, SAR and sensor weight, weapon weight, number of UAVs, number of SAR, number of sensors, binary decision variables, and integer decision variables.

3.4.1 Optimization Assumptions

Several assumptions have been made in the development of the MINLP. The assumptions include:

1. In order to scope the problem, all UAVs will have the same SAR, sensor, and weapon configurations.
2. In order to meet aircraft design standards, the total number of EO/IR sensors allowed on the aircraft is three and the total number of SAR allowed on the aircraft is one.
3. Because sensor packages represent “one of the single largest cost items in an unmanned aircraft” (“Unmanned Aircraft Systems Roadmap” 2005), only SAR and EO/IR sensors will be considered in the budget constraint.
4. Because weapons are given a specific weight limitation, weapon weight will be considered separately from the SAR and sensor weight. Additionally, weapons will be included on the aircraft such that there is minimal slack in the weapon weight constraint.
5. Because weapon weight is treated differently than SAR and sensor weight, weapons will not be considered in the size constraint.

3.4.2 Notation

The notation for the optimization formulation is listed below.

Sets

$I = \{1,\ 2,\ ..., n\}$ is the SAR type

$J = \{1, 2, ..., m\}$ is the sensor type

$K = \{1, 2, ..., l\}$ is the weapon type

Decision Variables

$$v_i = \begin{cases} 1, \text{ if SAR type } i \in I \text{ is used} \\ 0, \text{ otherwise} \end{cases}$$

$$w_j = \begin{cases} 1, & \text{if sensor type } j \in J \text{ is used} \\ 0, & \text{otherwise} \end{cases}$$

$x_k =$ number of weapon type $k \in K$ used

$y =$ number of UAVs

Parameters

$\alpha_i =$ cost associated with SAR type $i \in I$

$c_j =$ cost associated with sensor type $j \in J$

$s_i =$ size (ft^3) of SAR type $i \in I$

$\delta_j =$ size (ft^3) of sensor type $j \in J$

$\gamma_i =$ weight (lb) associated with SAR type $i \in I$

$\beta_j =$ weight (lb) associated with sensor type $j \in J$

$b =$ budget allowance for a single UAV

$S =$ size allowance (ft^3) for a single UAV

$W =$ SAR and sensor weight allowance (lb) for a single UAV

$A =$ weapon weight allowance (lb) for a single UAV

$B =$ total budget allowance for all UAVs

Simulation Output

$T_{kill} =$ number of targets killed

$T_{total} =$ total number of targets

3.4.3 Optimization Formulation

The MINLP associated with the simulation is described below. Define the following maximization problem:

$$\text{Max } z = \frac{T_{kill}}{T_{total}} \tag{3.1}$$

Subject to:

$$\sum_{i=1}^{|I|} \alpha_i v_i + \sum_{j=1}^{|J|} c_j w_j \le b \tag{3.2}$$

$$\sum_{i=1}^{|I|} s_i v_i + \sum_{j=1}^{|J|} \delta_j w_j \le S \tag{3.3}$$

$$\sum_{i=1}^{|I|} \gamma_i v_i + \sum_{j=1}^{|J|} \beta_i w_i \le W \tag{3.4}$$

$$\sum_{k=1}^{|K|} \omega_k x_k \le A \tag{3.5}$$

$$y\sum_{i=1}^{|I|} \alpha_i v_i + y\sum_{j=1}^{|J|} c_j w_j \le B \tag{3.6}$$

$$\sum_{i=1}^{|I|} v_i \le 1 \tag{3.7}$$

$$\sum_{j=1}^{|J|} w_j \le 3 \tag{3.8}$$

$$v_i \text{ and } w_{\mathrm{j}} \in [0,1] \ \forall \ i \in I, j \in J \tag{3.9}$$

$$x_k \text{ and } y \in \text{integer} \ \forall \ k \in K \tag{3.10}$$

As previously stated, the goal of the objective function, Equation (3.1), is to maximize the percentage of targets destroyed. The objective function value is obtained by running 30 replications of the simulation. The budget constraint, Constraint (3.2), sums the costs of the SAR and sensors used on the aircraft platform. The cost must be less than or equal to the allowed budget for one UAV, denoted by *b*. Constraint (3.3) provides the size constraint. The size constraint determines whether or not the SAR and sensors included on the platform are too large for the aircraft. The size constraint does not take into account the physical shape of the aircraft but, rather, provides a general guideline for whether or not a SAR or sensor should be included on the platform because of its size. The SAR and sensor weight constraint, Constraint (3.4), sums the weights of the SAR and sensors to be included on the aircraft. The combined weight of the SAR and sensors should be less than or equal to the given weight allowance for a single UAV, defined as W.

As noted in the list of assumptions, the weapons have a separate weight constraint, provided in Constraint (3.5). Here, the sum of the weapons included on the aircraft should be less than or equal to the weapon weight allowance for a single UAV, denoted by A. Constraint (3.6) is a nonlinear constraint that limits the total number of UAVs. The number of UAVs, y, is multiplied by the cost of the SAR and sensors included on the platforms. The cost should be less than or equal to the total allotted budget for all UAVs, defined by B. Constraint (3.7) limits the number of SAR to one, and Constraint (3.8) limits the number of sensors to three. Constraint (3.9) states that the decision variables for SAR and sensors are binary. Therefore, if a SAR or sensor type is included on the platform, it is represented as a one. If a SAR or sensor type is not included, it is represented as a zero. Constraint (3.10) states that the decision variables for number of weapon types and number of UAVs are integer.

3.5 Tabu Search Algorithm

A tabu search algorithm is used to perform the optimization portion of the simulation optimization procedure. The pseudo-code for the tabu search algorithm applied to the Rev/HK problem is provided in Figure 3-8.

```
• Generate a feasible starting solution, C, containing a EO/IR sensor set, SAR, weapons
  distribution, and the total number of UAVs
• Set the initial best solution to C* = C
• Calculate the percentage killed for C by running 30 simulation replications
• Add initial solution to elite candidate list
• FOR a set number of iterations
    o Choose a random number (rnd)
    o IF rnd ≤ Percentage determined through testing
        ▪ Perform a diversification move ("Multi-Swap Move") and determine a new
          feasible solution C' that has a new set of sensors, SAR, weapons
          distribution, and number of UAVs
    o ELSE
        ▪ Perform an intensification move ("Single Swap Move") on a member of the
          elite candidate list and determine a new feasible solution C' that has one
          different sensor
    o END IF
    o Determine the percentage killed for C' by running the simulation with replications
    o IF the percentage killed for C' < the percentage killed for C*
        ▪ IF the tabu list is empty
            • Set C = C'
            • Add the new elements to the tabu list
        ▪ ELSE
            • Determine if any of the SAR or sensor types are tabu
            • IF none of the elements are tabu
                o Set C = C'
                o Add the new SAR and sensor types to the tabu list
            • END IF
        ▪ END IF
    o ELSE
        ▪ Set C = C'
        ▪ Add new SAR and sensor types to the tabu list
        ▪ Set C* = C
        ▪ Add C to the elite candidate list
    o END IF
    o After tabu tenure
        ▪ Update tabu list
• END LOOP
• Return C*, the percentage killed associated with C*, and the run time associated with C*
• Return the values associated with the weight, budget, and size constraints for the given C*
```

Figure 3-8: Tabu search pseudo-code

3.5.1 Initial Input Parameters

In order to generate an initial parameter set, the tabu search algorithm calls a function named "initial parameters." First, "initial parameters" randomly chooses a set of sensors to consider. The function then addresses the feasibility of the sensors chosen independently. If a chosen sensor is feasible, it is added to the initial set of parameters until at most three sensors are added to the set. Next, the function randomly chooses a set of SARs to consider. Again, a feasibility check is required. Note that at most one SAR can be included in the initial set of input parameters because of the constraint limiting the number of SARs to one.

Additionally, the function randomly determines the weapons configuration. According to assumption four provided in Section 3.4.1, there should be minimal slack in the weapon weight constraint. Assume that there are three weapon types: x_1, x_2, and x_3. To determine the weapon distribution for the largest weapon type, x_1, the function first draws a random number. Next the function multiplies the random number by the total allowable weapon weight (A) and divides by the technological coefficient of the weapon type (ω_1). If this result is not an integer value, it is rounded down to the next integer value. This integer value represents the number of weapon type x_1 that are added to the platform. Next, the weapon weight (A) is decremented by the weight associated with the weapons added. This procedure is repeated for the second largest weapon type, x_2. The number of weapons for the last weapon type (x_3) is determined by dividing the remaining weapon weight (A) by the corresponding technological coefficient (ω_3) and rounding down to the nearest integer value. Section 3.7.1 provides an example of these calculations.

Finally, the "initial parameter" function randomly determines the number of UAVs to be used in the simulation. First, the function determines the maximum number of UAVs that the total budget (B) can sustain. The maximum is the floor of the total budget divided by the original budget minus the remaining budget. The function draws a random integer between one and the maximum value. The initial input parameter set is presented as an array with binary values representing the sensor and SAR types, where a one represents a sensor or SAR that is in use and a zero represents that a sensor or SAR that is not in use. The array also contains integers representing the number of each weapon type and number of UAVs.

3.5.2 Elite Candidate List

Each set of input parameters is evaluated by the simulation to provide an objective function value. The elite candidate list stores the initial set and its objective function value as well as additional parameter sets that have improving objective function values. The elite candidate list holds a finite number of sets (which will be determined through experimentation); therefore, if there are more improving sets than locations on the list, the newer sets will overwrite the older sets. The elite candidate list is used in the intensification portion of the algorithm.

3.5.3 Intensification

The tabu search algorithm calls the "single-swap move" function to perform the intensification process and develop a new set of input parameters. The intensification process occurs a certain percentage of the time, which is determined through experimentation. The algorithm randomly chooses a member of the elite candidate list to pass into the "single-swap move" function. The function performs one feasible sensor swap, i.e. the function exchanges a sensor currently in use for another feasible sensor, while maintaining the SAR configuration and the weapons distribution. The number of UAVs remains the same unless the total budget (*B*) cannot support the number of UAVs from the elite candidate list solution. If this is the case, the number of UAVs is changed to the maximum number of UAVs the budget can support.

3.5.4 Diversification

The tabu search algorithm calls the “multi-swap move” function to perform the diversification process and develop a new set of input parameters. Similarly to the intensification process, the diversification process occurs a certain percentage of the time, which will be determined through experimentation. The “multi-swap move” changes all of the configurations including the sensor set, the SAR, the weapons distribution, and the number of UAVs. Additionally, the “multi-swap move” can vary the number of sensors included on the platform between one and three.

Consider a solution created by the “initial parameter” function; the “multi-swap function” removes all sensors and SAR restoring the budget, size, and weight constraints to the original values. Additionally, the values for the weapons distribution and number of UAVs are cleared. The “multi-swap” function first randomly chooses the number of sensors to include on the platform, whereas the “initial parameter” function always includes three sensors on the platform. Next, the function randomly chooses different feasible sensors and SAR to include on the platform. The “multi-swap function” also determines a different weapons distribution and number of UAVs using the methods described in Section 3.4.1.

3.5.5 Tabu List

The tabu list holds sensors and SAR used on the platform in the recent past. The tabu tenure of a sensor or SAR is based upon the problem size. A separate list, tabu time, keeps track of how long a sensor or SAR has been on the tabu list. If the tabu time for a sensor or SAR equals the tabu tenure, the sensor or SAR is removed from the tabu list. Additionally, the tabu list contains the weapons configuration and the number of UAVs from the previous solution.

A configuration containing a tabu sensor or SAR value and/or the same weapons configuration or number of UAVs as the previous solution will not be considered as a possible solution by the algorithm. However, the aspiration criterion allows a configuration with the best objective function value yet evaluated to be considered as a possible solution.

3.6 Testing Component

DOE is used to develop a robust simulation optimization algorithm. Three tabu search parameters are chosen as factors in the design. The factors are chosen such that any aspect of the tabu search procedure that requires an analyst decision is tested. The DOE factors include: (1) the maximum number of iterates, (2) the intensification/diversification percentage, and (3) the elite candidate list length. Each factor will have a low level, a center point, and a high level.

A 2^3 factorial is used with blocking, where each block represents a different conflict scenario. The conflict scenarios are input into the simulation through the scenario inputs described in Section 3.3.1. These inputs, especially, weather, AOI, and number and type of targets, define the problem size. The responses recorded for the DOE include the objective function value, or percentage of targets destroyed, and the run time for the scenario. After the robust tabu search parameter settings are determined, simulation scenarios are run to determine the best input parameters.

3.7 Example Problem

In order to illustrate fully different aspects of the methodology, an example problem is described. This example problem represents a single replication of the DOE with robust parameter settings. The example problem consists of seven types of sensors, five types of SAR, and three types of weapons. The notation provided in the example problem corresponds to that described in the optimization component in Section 3.4.2. The three weapon types have associated weights (ω_k) of 500, 250, and 60 lb. Table 3-1 presents the technological coefficients associated with the five types of SAR.

Table 3-1: Example SAR technological coefficients

	α_i	s_i	γ_i
i	Cost	Size	Weight
1	\$40,000	3 ft^3	700 lb
2	\$20,000	2 ft^3	350 lb
3	\$25,000	2 ft^3	500 lb
4	\$50,000	3 ft^3	800 lb
5	\$35,000	2 ft^3	600 lb

Table 3-2 provides the technological coefficients for the seven sensor types.

Table 3-2: Example sensor technological coefficients

	c_j	δ_j	β_j
j	Cost	Size	Weight
1	\$25,000	2 ft^3	200 lb
2	\$30,000	2 ft^3	300 lb
3	\$20,000	1 ft^3	100 lb
4	\$15,000	1 ft^3	50 lb
5	\$35,000	3 ft^3	350 lb
6	\$40,000	3 ft^3	400 lb
7	\$27,500	2 ft^3	250 lb

Table 3-3 provides the right-hand side values of the constraints.

Table 3-3: Example right-hand side values

Right-Hand Side	Description	Value
b	Budget for one UAV	\$100,000
S	Size allowance for one UAV	10 ft^3
W	SAR/sensor weight for one UAV	2000 lb
A	Weapon weight for one UAV	1500 lb
B	Total cost for all UAVs	\$500,000

The algorithm requires both scenario and configuration inputs. Table 3-4 provides the example scenario inputs.

Table 3-4: Example scenario inputs

Scenario Parameter	Input
AOI	20 nmi by 10 nmi
Target Density	0.2 targets per nmi^2
% of Targets that are Vehicles	100%
Weather Conditions	scattered showers
Maximum UAV Altitude	40,000 ft
Maximum Speed	400 knots

The initial configuration inputs are determined through the use of the “initial parameters” function in the algorithm.

3.7.1 Example Initial Parameter Set

The function "initial parameters" might choose to consider sensors 4, 7, and 2 (provided in Table 3-2). The function would first assess sensor 4 for feasibility. Because sensor 4 meets the budget, weight, and size feasibility requirements, it is added to the initial solution, and the right-hand side values are decremented by the technological constraints associated with sensor 4. The process is repeated for sensors 7 and 2. Because both of the sensors meet the feasibility requirements, the sensors are also added to the initial solution. The remaining budget (b), size (S), and weight (W) values are now \$27,500, 5 ft^3, and 1400 lb, respectively.

Next, "initial parameter" attempts to add a SAR to the configuration. Given the example problem, "initial solution" might choose SAR type 2 (Table 3-1). SAR type 2 undergoes the budget, size, and weight feasibility checks successfully. Therefore, SAR type 2 is added to the initial solution and the remaining values for budget, size, and weight are \$7,500, 3 ft^3, and 1050 lb.

In order to determine the weapon distribution for the configuration, "initial parameter" follows the procedure outlined in section 3.4.1. For the given example, let the random numbers be 0.50 and 0.20. To determine the number of 500 lb weapons to be included on the aircraft, the following calculation is made:

$$\text{Number of 500 lb Weapons} = \left\lfloor \left(\frac{0.5*1500\text{ lb}}{500\text{ lb}}\right) \right\rfloor = 1.$$

Next, the number of 250 lb weapons is determined using the following calculation:

$$\text{Number of 250 lb Weapons} = \left\lfloor \left(\frac{0.2*1000\text{ lb}}{250\text{ lb}}\right) \right\rfloor = 0.$$

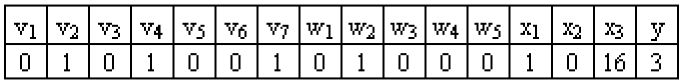
Finally, the number of 60 lb weapons is determined by the following calculation:

$$\text{Number of 60 lb Weapons} = \left\lfloor \frac{1000 \text{ lb}}{60 \text{ lb}} \right\rfloor = 16.$$

Recall that the maximum number of UAVs is calculated by taking the floor of the total budget divided by the original budget minus the remaining budget. In the case of the given example, the maximum number of UAVs is calculated as follows:

$$\text{Maximum Number of UAVs} = \left\lfloor \frac{\$500000}{\$100000 - \$7500} \right\rfloor = 5.$$

The number of UAVs is randomly set between 1 and the maximum; therefore, for the example, the number of UAVs (*y*) is set to 3. Figure 3-10 provides the initial parameter set for the example.

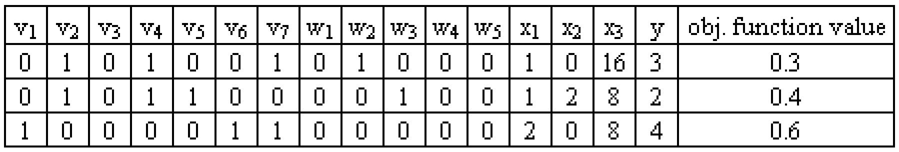


v_1	v_2	v_3	v_4	v_5	v_6	v_7	w_1	w_2	w_3	w_4	w_5	x_1	x_2	x_3	y
0	1	0	1	0	0	1	0	1	0	0	0	1	0	16	3

Figure 3-10: Example initial parameter set

3.7.2 Example Elite Candidate List and Intensification

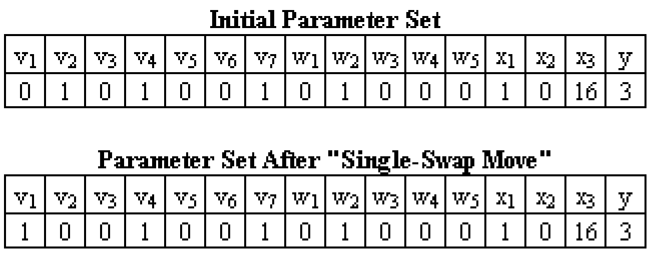
To determine the objective function value associated with a input parameter set, the simulation is run for 30 replications. The initial parameter set is stored in the elite candidate list. Additionally, parameter sets with improving solutions are also stored in the elite candidate list. Figure 3-11 provides an example elite candidate list that holds three solutions.



v_1	v_2	v_3	v_4	v_5	v_6	v_7	w_1	w_2	w_3	w_4	w_5	x_1	x_2	x_3	y	obj. function value
0	1	0	1	0	0	1	0	1	0	0	0	1	0	16	3	0.3
0	1	0	1	1	0	0	0	0	1	0	0	1	2	8	2	0.4
1	0	0	0	0	1	1	0	0	0	0	0	2	0	8	4	0.6

Figure 3-11: Example elite candidate list

In order to perform intensification and create a new parameter set, let the first solution in the elite candidate list be passed into the "single-swap move" function. The function randomly chooses a sensor to delete from the configuration, such as sensor 2, and updates the budget, size, and weight constraints. The “single-swap move” function then randomly chooses a new sensor to add to the configuration, such as sensor 1. Because sensor 1 is not currently in the sensor set and meets all of the feasibility requirements, it is added to the new solution. Figure 3-12 compares the two solutions before and after the “single-swap move” is performed.



Initial Parameter Set

v_1	v_2	v_3	v_4	v_5	v_6	v_7	w_1	w_2	w_3	w_4	w_5	x_1	x_2	x_3	y
0	1	0	1	0	0	1	0	1	0	0	0	1	0	16	3

Parameter Set After "Single-Swap Move"

v_1	v_2	v_3	v_4	v_5	v_6	v_7	w_1	w_2	w_3	w_4	w_5	x_1	x_2	x_3	y
1	0	0	1	0	0	1	0	1	0	0	0	1	0	16	3

Figure 3-12: Example “single-swap move”

3.7.3 Example Diversification

The diversification process creates a new set of parameters that is not intentionally related to any parameter set developed earlier in the algorithm. The diversification process determines the number of sensors to include on the aircraft. For the example, two sensors, such as sensor 6 and 3 (provided in Table3-2), are chosen to be included on the aircraft. Because both sensors are feasible, sensors 6 and 3 are added to the solution configuration and the budget, size, and weight constraints are updated to \$40,000, 5 ft^2, and 1300 lb, respectively.

The diversification also attempts to add a different SAR to the aircraft, for example, SAR type 3 (provided in Table 3-1). SAR type 3 meets the budget, size, and weight constraints; therefore, SAR type 3 is added to the solution set. The budget, size, and weight constraints are updated to \$15,000, 2 ft^2, and 800 lb, respectively.

For the example, the weapon distribution is changed to two 500 lb weapons and eight 60 lb weapons. The total budget can support 5 UAVs; therefore, the algorithm randomly chooses a value of 4 UAVs. Figure 3-13 provides an example comparison of the initial parameter configuration and the parameter configuration created by the "multi-swap move."

Initial Parameter Set

v_1	v_2	v_3	v_4	v_5	v_6	v_7	w_1	w_2	w_3	w_4	w_5	x_1	x_2	x_3	y
0	1	0	1	0	0	1	0	1	0	0	0	1	0	16	3

Parameter Set After "Multi-Swap Move"

v_1	v_2	v_3	v_4	v_5	v_6	v_7	w_1	w_2	w_3	w_4	w_5	x_1	x_2	x_3	y
0	0	1	0	0	1	0	0	0	1	0	0	2	0	8	4

Figure 3-13: Example "multi-swap move"

3.7.4 Example Tabu List

Figure 3-14 provides an example tabu list after the "initial parameters" and the "multi-swap move" functions are performed. Additionally, the figure presents the tabu time list that keeps track of how long a sensor or SAR has been on the tabu list. Assume the tabu tenure is greater than two iterations.

Tabu List After Initial Set and "Multi-Swap Move"

v_1	v_2	v_3	v_4	v_5	v_6	v_7	w_1	w_2	w_3	w_4	w_5	x_1	x_2	x_3	y
0	1	1	1	0	1	1	0	1	1	0	0	2	0	8	4

Tabu Time List

v_1	v_2	v_3	v_4	v_5	v_6	v_7	w_1	w_2	w_3	w_4	w_5
0	2	1	2	0	1	2	0	2	1	0	0

Figure 3-14: Example tabu list and tabu time list

The algorithm runs until it reaches a maximum number of iterations. It then outputs the configuration with the highest objective function value, the maximum objective function value, and the time required to determine the objective function value.

3.8 Conclusion

Simulation optimization is a technique that can be used to determine the best set of input parameters for a very complex system that is modeled via simulation. This chapter provided a methodology that applied simulation optimization to the Rev/HK problem. The simulation model of the Rev/HK flight and military actions was described in detail. Additionally, the optimization formulation associated with the simulation was discussed. This formulation was used to develop a tabu search metaheuristic that was tied into the simulation. A testing procedure was outlined in order to develop a robust simulation optimization algorithm. Finally, an example problem was provided to demonstrate the previously described methodology. Chapter 4 will provide the results and analysis of the implementation of this methodology.

IV. Results and Analysis

4.1 Introduction

Simulation optimization provides a method to find the optimal or near optimal input parameters to complex simulations. Simulation optimization can be applied to many types of problems including manufacturing, financial planning, and workforce planning. The methodology presented in Chapter 3 provided a simulation optimization procedure to solve a resource allocation problem, specifically the Rev/HK problem. This chapter will briefly review that methodology. Additionally, it will next provide the designed experiment developed to test the methodology. The results and analysis of this designed experiment are also provided.

4.2 Simulation Optimization Applied to the Rev/HK Problem

The simulation optimization methodology developed in Chapter 3 involved a simulation that described the flight and military patterns of a Rev/HK UAV and a tabu search optimizer. The goal of the simulation optimization procedure is to provide a set of input parameters to the simulation that maximizes the objective function, which is the percentage of targets destroyed. The tabu search optimizer serves primarily as a way to check the feasibility of the input parameters and to store the input parameters that achieve the best objective function value. The optimizer can either try to improve the objective function value by intensifying around input parameter sets stored in memory that have provided large objective function values in the past, or the optimizer can diversify by exploring new parameter settings.

The methodology provided in Chapter 3 first uses the tabu search optimizer to develop a feasible set of input parameters, specifically a set of EO/IR sensors, a SAR, a weapons distribution, and the total number of UAVs, to use in the simulation. The input parameters must meet budgetary, size, and weight constraints for each UAV and a budgetary constraint for the UAV fleet as a whole. Thirty simulation replications are run using the input parameters provided by the tabu search and the percentage of targets destroyed and the simulation run time are recorded. Next, the procedure is repeated using a new set of feasible input parameters provided by the tabu search optimizer. The process continues until a maximum number of iterations is reached. Figure 4-1 outlines the simulation optimization procedure applied to the Rev/HK problem.

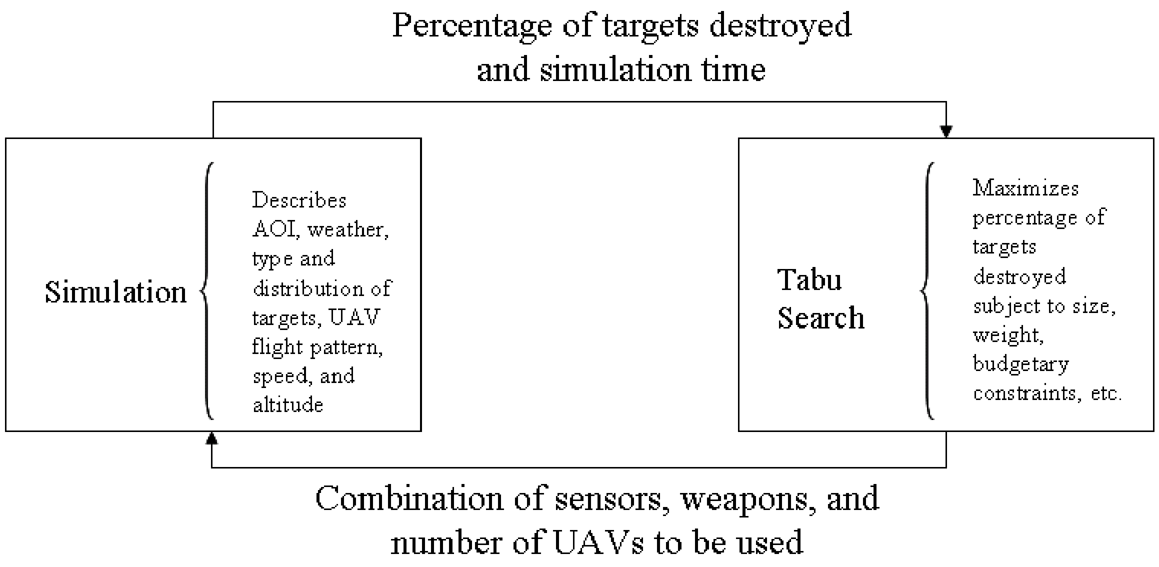

Percentage of targets destroyed and simulation time
Simulation
Describes AOI, weather, type and distribution of targets, UAV flight pattern, speed, and altitude
Tabu Search
Maximizes percentage of targets destroyed subject to size, weight, budgetary constraints, etc.
Combination of sensors, weapons, and number of UAVs to be used

Figure 4-1: Simulation optimization applied to the Rev/HK problem

4.3 Data

In order to test the simulation optimization methodology, cost, weight, and size data relating to fifteen sensors and 7 SARs are collected. This data is used in the methodology to determine the feasibility of different sensor and SAR combinations.

Additionally, characteristics relating to the sensor and SAR functionality, specifically the sensor or SAR resolution, are recorded. Table 4-1 provides the technological coefficients and resolution associated with each SAR. SAR resolution improves as the resolution value decreases; therefore, SAR 3 has the best resolution (1 ft), while SAR 4 has the worst resolution (5 ft).

Table 4-1: SAR technological coefficients and resolution

	α_i	s_i	γ_i	
i	**Cost**	**Size**	**Weight**	**Resolution**
1	\$6.3 M	0.87 ft^3	450 lb	3 ft
2	\$6.7 M	0.91 ft^3	400 lb	2 ft
3	\$7.6 M	0.90 ft^3	350 lb	1 ft
4	\$4.9 M	0.97 ft^3	600 lb	5 ft
5	\$5.9 M	1.16 ft^3	510 lb	3 ft
6	\$5.4 M	1.10 ft^3	500 lb	4 ft
7	\$7.1 M	0.93 ft^3	380 lb	2 ft

Table 4-2 provides the technological coefficients associated with the different sensor types. Additionally, the table provides the resolution associated with each sensor. Sensor resolution improves as the resolution value increases. Therefore, sensors 1, 8, and 15 have the best resolution (1280 x 720), while sensors 3, 6, and 14 have the worst resolution values (360 x 240).

Table 4-2: Sensor technological coefficients and resolution

	c_j	δ_j	β_j	
j	Cost	Size	Weight	Resolution
1	\$1 M	3.05 ft^3	136 lb	1280 x 720
2	\$0.2 M	3.05 ft^3	136 lb	640 x 480
3	\$0.13 M	0.86 ft^3	60 lb	320 x 240
4	\$0.16 M	2.39 ft^3	80 lb	640 x 480
5	\$0.33 M	2.96 ft^3	150 lb	640 x 480
6	\$0.09 M	1.55 ft^3	80 lb	320 x 240
7	\$0.35 M	2.45 ft^3	100 lb	640 x 480
8	\$1.5 M	2.89 ft^3	120 lb	1280 x 720
9	\$0.5 M	3.12 ft^3	140 lb	1024 x 768
10	\$0.6 M	2.82 ft^3	127 lb	1024 x 768
11	\$0.275 M	2.85 ft^3	130 lb	640 x 480
12	\$0.15 M	3.16 ft^3	141 lb	640 x 480
13	\$0.23 M	2.72 ft^3	100 lb	640 x 480
14	\$0.2 M	0.65 ft^3	45 lb	320 x 240
15	\$0.9 M	3.54 ft^3	150 lb	1280 x 720

Table 4-3 presents the right-hand side values for the constraints.

Table 4-3: Right-hand side values

Right-Hand Side	Description	Value
b	Budget for one UAV	\$9 M
S	Size allowance for one UAV	10 ft^3
W	SAR/sensor weight for one UAV	1000 lb
A	Weapon weight for one UAV	1500 lb
B	Total cost for all UAVs	\$40 M

Additionally, three weapon types are included in the model, with the associated weights of 500, 250, and 60 lb.

4.4 Designed Experiment

In order to ensure the methodology developed in Chapter 3 is robust, design of experiments (DOE) is used to determine the appropriate tabu search parameters. Three tabu search parameters, (1) the intensification/diversification percentage, (2) the elite candidate list length, (3) and the maximum number of iterates, are factors in the experiment and are tested at low, center point, and high levels. Three replicates of a full 2^3 factorial design with 12 blocks, where each block represents a conflict scenario, are performed.

4.4.1 Factor Levels

The three tabu search parameters that are viewed as factors in the DOE are tested at low, center point, and high levels. The low, center point, and high intensification/diversification percentage levels are 75%, 87.5%, and 100%, respectfully. These factor level values are chosen to test the whether or not the algorithm should only diversify or if the algorithm should include both intensification and diversification. Recall from Section 3.5, the tabu search procedure will perform diversification moves when the random number rnd is less than or equal to the intensification/diversification percentage. Therefore, a 100% intensification/diversification percentage ensures that the algorithm will only perform diversification moves. The 100% setting will allow a wider range of parameter settings to be tested. A broader range of parameter settings will allow more of the solution space to be searched, which may prove beneficial because of the computational time required to perform the simulation optimization procedure.

The elite candidate list length factor level values are chosen to be relatively small so that the solutions stored in the elite candidate list truly represent the best solutions generated by the algorithm. The elite candidate list length ranges from three at the low level to four at the center point to five at the high level. However, the elite candidate list will only affect the model if the intensification/diversification percentage is less than 100%.

The maximum number of iterates factor level values are chosen such that a sufficient number of tabu search iterations occur while maintaining a reasonable computational run time. The maximum number of iterates has a low level of 20, a center point of 25, and a high level of 30. The DOE factors and factor levels are presented in Table 4-4.

Table 4-4: Factor and factor levels for Rev/HK DOE

Factor	Low (-1)	Center (0)	High (1)
A: Intensification/Diversification Percentage	75%	87.5%	100%
B: Elite Candidate List Length	3	4	5
C: Maximum Number of Iterates	20	25	30

4.4.2 Blocking

The blocks in the DOE represent different simulation scenarios. The simulation scenarios consist of the following inputs: AOI, target characteristics, weather conditions, flight pattern, maximum altitude, and maximum speed. The scenario inputs, specifically, AOI, target characteristics, and weather conditions, are used to define the problem size. A total of twelve scenarios are developed by varying these inputs: the AOI can be small (5 nmi by 5 nmi), medium (10 nmi by 10 nmi), or large (15 nmi by 15 nmi); the weather can either be good (clear skies) or bad (rain); and the target density of vehicles can be either 20% of targets per nmi^2 or 50% of targets per nmi^2. Table 4-5 lists the 12 possible scenarios.

Table 4-5: Simulation scenarios

Scenario	AOI	Weather	Target Density
1	Small [5 5]	Good (clear)	0.2
2			0.5
3		Bad (rain)	0.2
4			0.5
5	Medium [10 10]	Good (clear)	0.2
6			0.5
7		Bad (rain)	0.2
8			0.5
9	Large [15 15]	Good (clear)	0.2
10			0.5
11		Bad (rain)	0.2
12			0.5

The remaining scenario inputs are the same for all scenarios. All scenarios use a zamboni, S-shaped, search pattern, a maximum altitude of 10,000 ft, and a maximum speed of 400 knots. Note the maximum altitude is less than the minimum service ceiling of 15,000 ft for the Rev/HK UAV. The choice of a lower altitude was made deliberately. Because the EO/IR sensors are attached to the UAV at an angle, a higher altitude would require a much larger AOI in order to include the areas being viewed by the sensors. Very large AOIs are much more computationally expensive than the AOIs tested; therefore, a lower altitude allowed for experimentation across AOIs of different sizes, while maintaining a reasonable simulation optimization run time.

Animation is employed to allow the analyst to view the simulation as it is run. Figure 4-2 presents the animation associated with the simulation using simulation scenario four. There is one UAV, denoted by *, that is equipped with one EO/IR sensor (the red box in front of the UAV) and one SAR (the pink box around the UAV). The UAV's path is marked by a dashed line. The large boxes within the AOI are clouds/rain and the small boxes are targets.

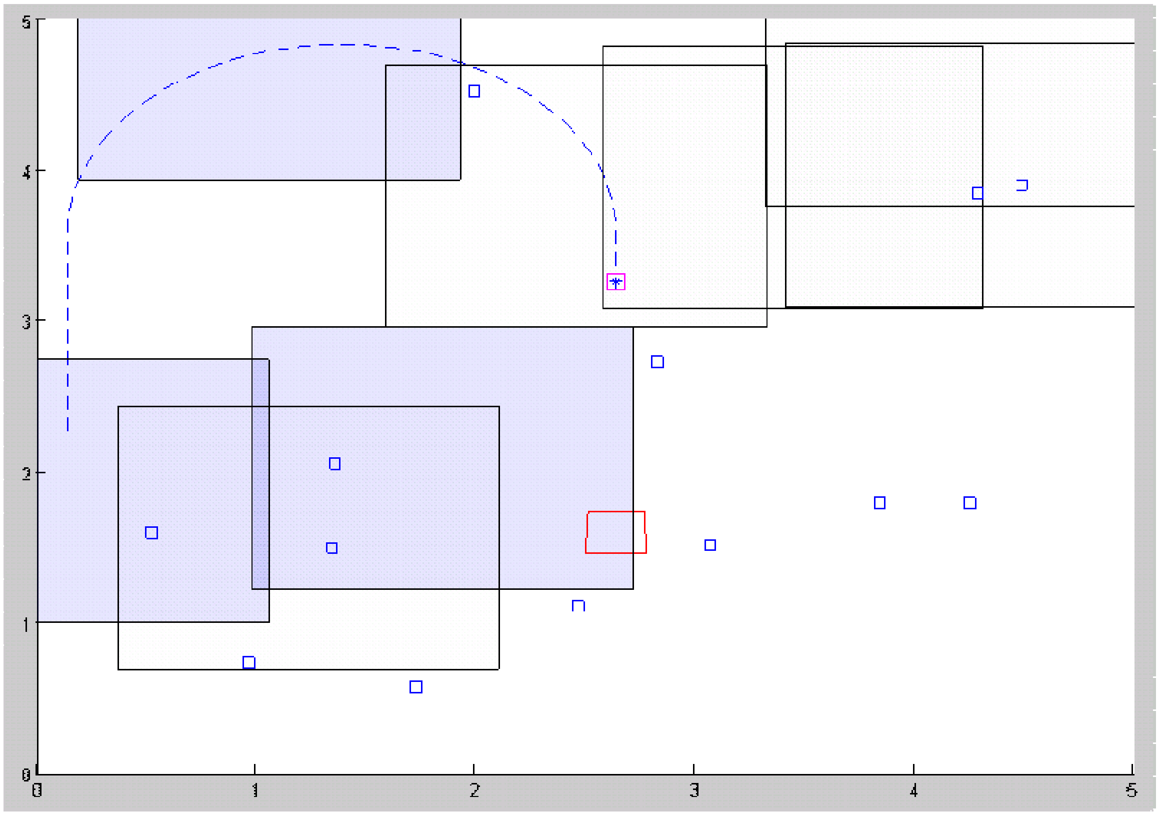


Figure 4-2: Example simulation animation

4.4.3 Experiment

A total of 36 runs are required to test the three factors at the low, center point, and high levels using 12 blocks. Table 4-6 presents the experiment.

Table 4-6: Design of experiments for the Rev/HK problem

		Factors		
Run	**Scenario**	**A**	**B**	**C**
1	Scenario 1	1	3	20
2	Scenario 1	0.875	4	25
3	Scenario 1	0.75	5	30
4	Scenario 2	1	5	20
5	Scenario 2	0.875	4	25
6	Scenario 2	0.75	3	30
7	Scenario 3	1	3	30
8	Scenario 3	0.75	5	20
9	Scenario 3	0.875	4	25
10	Scenario 4	0.75	3	20
11	Scenario 4	0.875	4	25
12	Scenario 4	1	5	30
13	Scenario 5	0.75	5	30
14	Scenario 5	1	3	20
15	Scenario 5	0.875	4	25
16	Scenario 6	0.875	4	25
17	Scenario 6	0.75	3	30
18	Scenario 6	1	5	20
19	Scenario 7	0.875	4	25
20	Scenario 7	1	3	30
21	Scenario 7	0.75	5	20
22	Scenario 8	0.75	3	20
23	Scenario 8	0.875	4	25
24	Scenario 8	1	5	30
25	Scenario 9	0.75	5	30
26	Scenario 9	1	3	20
27	Scenario 9	0.875	4	25
28	Scenario 10	1	5	20
29	Scenario 10	0.75	3	30
30	Scenario 10	0.875	4	25
31	Scenario 11	1	3	30
32	Scenario 11	0.875	4	25
33	Scenario 11	0.75	5	20
34	Scenario 12	0.875	4	25
35	Scenario 12	1	5	30
36	Scenario 12	0.75	3	20

4.5 DOE Results

Table 4-7 provides the results of the REV/HK DOE.

Table 4-7: DOE results

		Factors			% of Targets Killed	Simulation Run Time (sec)
Run	Scenario	A	B	C		
1	Scenario 1	1	3	20	20.00%	126.83
2	Scenario 1	0.875	4	25	13.33%	46.75
3	Scenario 1	0.75	5	30	14.67%	276.63
4	Scenario 2	1	5	20	25.13%	77.11
5	Scenario 2	0.875	4	25	22.05%	130.86
6	Scenario 2	0.75	3	30	40.51%	79.1406
7	Scenario 3	1	3	30	59.33%	167.2
8	Scenario 3	0.75	5	20	23.33%	117.4
9	Scenario 3	0.875	4	25	51.33%	43.47
10	Scenario 4	0.75	3	20	30.51%	325.17
11	Scenario 4	0.875	4	25	20.77%	147.63
12	Scenario 4	1	5	30	24.10%	98.78
13	Scenario 5	0.75	5	30	68.50%	359.59
14	Scenario 5	1	3	20	45.83%	248.98
15	Scenario 5	0.875	4	25	55.50%	149.7
16	Scenario 6	0.875	4	25	43.33%	490.9
17	Scenario 6	0.75	3	30	57.87%	579.38
18	Scenario 6	1	5	20	34.90%	485.37
19	Scenario 7	0.875	4	25	79.00%	386.36
20	Scenario 7	1	3	30	79.00%	376.75
21	Scenario 7	0.75	5	20	29.33%	596.45
22	Scenario 8	0.75	3	20	54.33%	1775.19
23	Scenario 8	0.875	4	25	11.47%	2515.8
24	Scenario 8	1	5	30	28.33%	698.48
25	Scenario 9	0.75	5	30	7.48%	888.65
26	Scenario 9	1	3	20	18.52%	892.16
27	Scenario 9	0.875	4	25	16.15%	1584.6
28	Scenario 10	1	5	20	21.45%	3506.3
29	Scenario 10	0.75	3	30	54.54%	1675.7
30	Scenario 10	0.875	4	25	21.59%	1677
31	Scenario 11	1	3	30	24.74%	1211.5
32	Scenario 11	0.875	4	25	23.41%	1068.2
33	Scenario 11	0.75	5	20	20.00%	1159.88
34	Scenario 12	0.875	4	25	28.08%	2796.5
35	Scenario 12	1	5	30	51.42%	2598.3
36	Scenario 12	0.75	3	20	23.25%	2622.6

An analysis of each the response is performed. The model for the percentage of targets killed is determined to be significant; however, the model for simulation run time is determined to be statistically insignificant. Figure 4-3 displays a half normal probability plot of the effects associated with the response percentage of targets killed. Note that factors, B (elite candidate list length), C (maximum number of iterations), and interaction AB are deemed significant, and, therefore, will be included in the model. Additionally, factor A (intensification/diversification percentage) must be included in the design for hierarchal purposes.

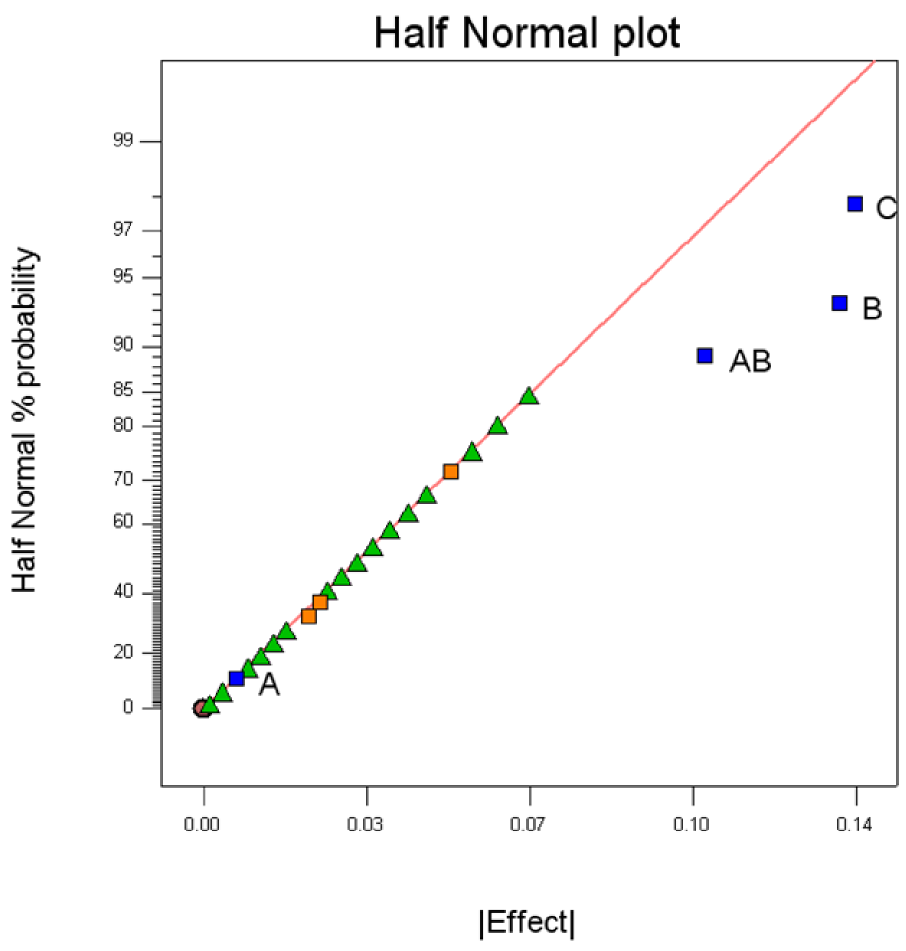


Figure 4-3: Half normal probability plot

Table 4-8 provides the analysis of variance (ANOVA) for the response percentage of targets killed. Note that the p-values highlighted in bold are all less than 0.05 and are associated with the significant factors (the model, B, C, and the interaction effect, AB).

Table 4-8: ANOVA for the percentage of targets killed

Source	Sum of Squares	DF	Mean Square	F Value	p - value
Block	0.755901589	11	0.068718326		
Model	0.284916164	4	0.071229041	5.63707295	**0.0036**
A	0.000295998	1	0.000295998	0.02342534	0.8800
B	0.106384853	1	0.106384853	8.419307198	**0.0091**
C	0.111945749	1	0.111945749	8.859397046	**0.0078**
AB	0.066289563	1	0.066289563	5.246162216	**0.0336**
Curvature	0.010046886	1	0.010046886	0.795111466	0.3837

Although the p-value for factor A is greater than 0.05, the factor is included in the model because of hierarchal purposes. Additionally, recall that factor A is the intensification/diversification percentage and factor B is the elite candidate list. As previously stated in Section 4.4.1, factor B is included in the model only if factor A is less than 100%. This relationship is a contributing factor to the significance of the interaction term.

The model for predicting the percentage of targets killed is

$$\hat{y} = 2.80616 - 2.88482 * \mathrm{A} - 0.70378 * \mathrm{B} + 0.013659 * \mathrm{C} + 0.72823 * \mathrm{AB}.$$

The R^2 and R^2-adjusted for the model are 0.5427 and 0.4464, respectively. Therefore, the model explains 44.64% of the system variability when adjusted for degrees of freedom. The relatively low adjusted R^2 value can be explained by the stochastic nature of the simulation optimization process.

In order to maximize the percentage of targets killed, numerical optimization using the prediction model determines that the factor levels A, B, and C are set to 75%, 3, and 30, respectively. These tabu search parameters are the robust parameter settings that will be used to develop the input parameters for different scenarios evaluated by the model. The robust tabu search parameter settings result in an algorithm that diversifies 75% of the time. However, the algorithm does intensify around one of the three elite candidate list parameter sets 25% of the time. Additionally, by setting the maximum number of iterations to 30, one ensures that the algorithm is allowed to perform more tabu search iterations throughout the solution space.

4.6 Input Parameter Settings

In order to demonstrate the ability of the algorithm to determine input parameter settings, scenarios one through four (presented in Section 4.4.2) are tested using the robust tabu search parameter settings. Table 4-9 presents the results of these tests, including the percent of targets killed, the simulation run time, and the input parameter settings.

Table 4-9: Results Using the Robust Tabu Search Parameter Settings

	Scenario 1	Scenario 2	Scenario 3	Scenario 4
% of Targets Killed	68.67%	44.36%	52.67%	69.74%
Simulation Run Time (sec)	45.7	94.09	42.31	305.62
Sensor Configuration	[15]	[8]	[1]	[8]
SAR Configuration	[4]	[7]	[7]	[4]
Weapons Distribution	[2,1,4]	[2,0,8]	[2,0,8]	[0,5,4]
Number of UAVs	8	4	4	5

Each input parameter set only has a single sensor. The sensors that are included on the aircraft are sensors 1, 8, and 15. Recall, from Section 4.3, that all of these sensors have the highest resolution available. Additionally, sensors 1, 8, and 15 are the most expensive sensors available; however, because only one sensor is used per UAV, the overall sensor cost per UAV may be less than the cost associated with including two or three sensors with lower resolution on a UAV.

The SAR included on the UAVs for scenarios one through four are SARs 4 and 7. SAR 4 has a resolution of 5 ft and a cost of \$4.9 M and SAR 7 has a resolution of 2 ft and a cost of \$7.1 M. The weapons distributions vary from two 500 lb weapons, one 250 lb weapon, and four 60 lb weapons for scenario one, to two 500 lb weapons and eight 60 lb weapons for scenarios two and three, to five 250 lb weapons to four 60 lb weapons for scenario four.

The number of UAVs also varies across the four scenarios. Scenario one requires eight UAVs. Scenarios two and three use four UAVs, and scenario four requires five UAVs.

4.7 Conclusion

The simulation optimization methodology applied to the Rev/HK problem incorporates a simulation of the Rev/HK UAV and a tabu search optimizer to determine the optimal or near optimal simulation input parameter settings. This chapter provided the data required to perform the testing component of the methodology described in Chapter 3. Additionally, the designed experiment, including the factor levels, the blocking component, and the layout of the experiment, was provided and discussed. The results of this experiment were presented and analysis, including a half normal probability plot and an ANOVA table, was provided. From this analysis, the robust tabu search parameter settings were determined. Finally, using the robust tabu search parameter settings, the input parameter settings for four simulation scenarios were determined. Chapter five provides concluding remarks and a discussion of future research opportunities.

V. Conclusion and Recommendations

5.1 Introduction

Simulation optimization is a methodology that combines two cornerstones of operations research: simulation and optimization. The goal of simulation optimization is to determine the simulation input parameters that provide optimal or near optimal simulation performance measures. Simulation optimization is currently applied to problems in manufacturing, workforce planning, facility layout and design, and financial planning (Glover et al. 1999).

This research provided a simulation optimization methodology to solve resource allocation problems. Specifically, this research addressed the allocation of EO/IR sensors, SAR, and weapons to the Rev/HK UAV. In order to apply the simulation optimization methodology to the Rev/HK problem, a simulation of the flight and military actions of the Rev/HK UAV was developed in MATLAB. A tabu search metaheuristic operated as the optimizer, providing the simulation with feasible input parameter sets. DOE was employed to develop a robust simulation optimization procedure.

5.2 Conclusion

This research effort addressed all objectives presented in Chapter 1. The objectives of this thesis are:

1. Develop a methodology that incorporates both simulation and optimization into a resource allocation problem.
2. Test the robustness of the methodology using design of experiments (DOE).

3. Determine the optimal or near optimal input parameters given specific simulation scenarios.

The first objective was addressed in Chapter 3. A simulation model describing the flight and military actions of the Rev/HK UAV was discussed in detail. Additionally, Chapter 3 presented an optimization formulation using the simulation output as the objective function and placing constraints on the simulation input parameters. A tabu search metaheuristic implemented the optimization formulation and tied together the simulation and optimization portions of the problem.

The methodology developed in Chapter 3, including the algorithms and computer code, can be used as a decision support tool for UAV development and design. Additionally, the methodology provides other researchers and analysts an example of how to apply simulation optimization to a resource allocation problem.

The second objective was addressed in Chapter 4. The data required to perform the designed experiment were provided. Additionally, the designed experiment, including the factors, factor levels, blocking component, and experiment layout, was described. The results of the designed experiments were reported and analysis was performed. The analysis resulted in determining the robust tabu search parameter settings. The development of robust tabu search design parameters demonstrates how researchers and analysts can successfully apply DOE to the simulation optimization framework.

The third and final objective was also addressed in Chapter 4. Using the robust tabu search parameter settings, the input parameter settings for four simulation scenarios were determined. The development of sensor, SAR, and weapon configurations can be used by the developers of the Rev/HK UAV when determining which sensors, SAR, and weapons to purchase and utilize. Additionally, the implementation of the algorithm can determine the size of the UAV fleet needed to achieve the optimal or near optimal percentage of targets killed.

5.3 Future Research

This research effort has not only addressed the research objectives proposed in Chapter 1 but has also provided additional research opportunities. First, the simulation model provided describes the actions of the Rev/HK UAV and the space in which the UAV operates. This space is described by the area of interest, the target characteristics, and weather conditions. Additional fidelity can be added to the simulation by adding aspects such as terrain and manmade structures to further describe the space in which the UAV operates. Increasing model fidelity will allow the simulation model to better describe the simulation scenario; however, increased fidelity will also increase the computational time required to complete the simulation optimization process.

Currently, the optimization component of the problem includes a budget constraint for each UAV. It is possible this constraint could be removed, allowing for the number of UAVs in the fleet to be reduced and the development of a UAV that includes the most advanced sensors and SAR available. Additionally, in order to scope the problem, the proposed model assumes that all UAVs in the fleet have the same sensor configuration, SAR configuration, and weapons distribution. However, it may be beneficial to develop a problem formulation that allows the UAVs in the fleet to have different sensor configurations, SAR configurations, and weapons distributions.

Appendix A. Tabu Search MATLAB Code

A.1 Tabu Search

```
%Perform tabu search

%load data
load('X.mat');
load('Y.mat');
load('NIIRS_data.mat');
load('SAR_data.mat');
load('SAR_truck_data.mat');
load('SAR_infantry_data.mat');

%Constraint Parameters
total_budget = 40; %budget for UAV fleet
budget = 9;  %budget for a single UAV
weight = 1000; %payload weight allowance
size = 10;  %payload size allowance
weapon_weight = 1500; %weapon weight allowance

%Tabu Search Parameters
div_per = 0.75; %percentage of the time requiring diversification
max_els = 3; %length of elite candidate list
MAX_IT = 30; %maximum number of iterations

%Simulation Parameters
area = [5 5];  %aoi configuration
weather_conditions = [0 0];  %weather configuration
search_pattern = 2;  %search pattern
max_alt = 10000;  %max altitude
max_speed = 400;  % max speed
targ = [.20 100];  % target configuration setup targets by:
                   [target_density percentage_vehicle]
sim_weapon_weight = 1500; %weapon weight allowance
k = 1;
elite_list_size = 1;
n = length(X);  %input X
m = length(Y);  %input Y
w = 3;  %number of weapon types
sensors = zeros(1,n);
SAR = zeros(1,m);
best_sensors = zeros(1,n);
current_sensors =zeros(1,n);
best_SAR = zeros(1, m);
current_SAR = zeros(1,m);
elite_list = zeros(max_els,n+m+w+2);
tabu = zeros(1,n+m+w+1);
tabu_time = zeros(1,n+m);
no_sensors = 0;
weapons = zeros(1, w);
best_weapons = zeros(1,w);
```

```
current_weapons = zeros(1,w);
weapon_tabu = zeros(1, w);
no_SAR = 0;
%create initial solution
[sensors, tabu, budget, weight, size, SAR, weapons, no_uav,
orig_budget, tabu_time] = initial_parameters(X, Y, budget, weight,
size, sensors, SAR, weapons, tabu, tabu_time, n, m, w, no_sensors,
no_SAR, weapon_weight, total_budget);

%evaluate initial feasible solution
best_sensors = sensors;
best_SAR = SAR;
current_sensors = sensors;
current_SAR = SAR;

%Run simulation
[pk, sim_time] = Run_simulation(area, weather_conditions,
search_pattern, max_alt, max_speed, targ, sim_weapon_weight, sensors,
SAR, weapons, n, m, SAR_data, NIIRS_data, SAR_infantry_data,
SAR_truck_data, no_uav);

current_pk = pk;
best_pk = pk;
best_time = sim_time;

%Add intial solution to elite list
elite_list(elite_list_size,(1:n))  = sensors;
elite_list(elite_list_size,(n+1:n+m)) = SAR;
elite_list(elite_list_size,(n+m+1: n+m+w)) = weapons;
elite_list(elite_list_size, n+m+w+1) = no_uav;
elite_list(elite_list_size, n+m+w+2) = pk;

for k = 2:MAX_IT

    tabu_item = 0;

    for i = 1:(n+m) %update the amount of time a sensor or SAR has been
                     on the tabu list
        if tabu_time(i) > 0
            tabu_time(i) = tabu_time(i)+1;
        end
    end

    if rand < div_per  %diversification
        [sensors, budget, weight, size, new_sens, r, SAR, new_SAR,
         weapons, no_uav] = multi_swap(X, Y, sensors, SAR, budget,
         weight, size, n, m, w, weapons, weapon_weight, no_uav,
         orig_budget, total_budget);
        type = 3;
    else  %intensification
        intensify = unidrnd(elite_list_size, 1, 1);
        sensors = elite_list(intensify,(1:n));  %Intensify around
                                                solution in elite
                                                candidate list
        SAR = elite_list(intensify,(n+1:n+m));
        weapons = elite_list(intensify, (n+m+1:n+m+w));
        no_uav = elite_list(intensify, (n+m+w+1));
        [sensors, budget, weight, size, new_sens, r, no_uav] =
         single_swap(X, sensors, budget, weight, size, n, orig_budget,
         total_budget, no_uav);
        type = 1;
    end

%Run simulation
    [pk] = Run_simulation(area, weather_conditions, search_pattern,
     max_alt, max_speed, targ, sim_weapon_weight, sensors, SAR, weapons,
     n, m, SAR_data, NIIRS_data, SAR_infantry_data, SAR_truck_data,
     no_uav);

    if pk < best_pk  %aspiration criteria not met
        if isequal(tabu(1:n+m), zeros(1,n+m))
            current_sensors = sensors; %New sensor set is not tabu and
                                        set to the current sensor set
            current_SAR = SAR;  %New SAR set is not tabu and set to the
                                 current SAR set
            current_weapons = weapons;  %New weapons set is set to the
                                         current weapons set
            current_pk = pk;

            for j = 1:r
                tabu(new_sens(j)) = 1;  %Set new sensors to tabu
                tabu_time(new_sens(j)) = 1; %Set tabu time to 1
            end
            if type == 3
                tabu(n + new_SAR(1)) = 1;  %Set new SAR to tabu
                tabu_time(n + new_SAR(1)) = 1; %Set tabu time to 1
            end
            for l = 1:w
                tabu(n+m+l) = weapons(l);  %Set new weapons set to tabu
            end
            tabu(n+m+w+1)= no_uav; %Set number of UAVs to tabu
        else
            for i = 1:r
                if tabu(new_sens(i)) == 1  %New sensor set is tabu
                    tabu_item = 1;
                    break
                end
            end
            if type == 3
                if tabu(n + new_SAR(1)) == 1  %New SAR is tabu
                    tabu_item = 1;
                end
            end
            if tabu_item == 0
                current_sensors = sensors;  %New sensor set is not tabu
                current_SAR = SAR;  %New SAR is not tabu
                current_weapons = weapons;  %New weapons set is set to
                                             the current weapons set
                for j = 1:r
                    tabu(new_sens(j)) = 1;  %Set new sensors to tabu
                    tabu_time(new_sens(j)) = 1; %Set tabu time to 1
                end
                if type == 3
                    tabu(n + new_SAR(1)) = 1;  %Set new SAR to tabu
                    tabu_time(n + new_SAR(1)) = 1; %Set tabu time to 1
                end
                for l = 1:w
                    tabu(n+m+l) = weapons(l);  %Set new weapons set to
                                                 tabu
                end
                tabu(n+m+w+1)= no_uav; %Set number of UAVs to tabu
            end
         end
    else %aspiration criteria met

        current_sensors = sensors;
        current_SAR = SAR;
        current_weapons = weapons;
        best_sensors = sensors;  %New sensor set provides an
                                 improvement
        best_SAR = SAR;  %New SAR provides an improvement
        best_weapons = weapons;  %New weapons provides an improvement
        last_best = best_pk;
        elite_list_size = elite_list_size + 1;

        %Update elite list
        if elite_list_size <= max_els
            elite_list(elite_list_size,(1:n))  = current_sensors;
            elite_list(elite_list_size,(n+1:n+m)) = current_SAR;
            elite_list(elite_list_size,(n+m+1:n+m+w)) =
            current_weapons;
            elite_list(elite_list_size,n+m+w+1) = no_uav;
            elite_list(elite_list_size,n+m+w+2) = pk;
        else
            elite_list_size = 1;
            elite_list(elite_list_size,(1:n))  = current_sensors;
            elite_list(elite_list_size,(n+1:n+m)) = current_SAR;
            elite_list(elite_list_size,(n+m+1:n+m+w)) =
            current_weapons;
            elite_list(elite_list_size,n+m+w+1) = no_uav;
            elite_list(elite_list_size,n+m+w+2) = pk;
        end
        best_pk = pk;
        best_time = sim_time;
        for j = 1:r
            tabu(new_sens(j)) = 1;  %Set new sensors to tabu
            tabu_time(new_sens(j)) = 1; %Set tabu time to 1
        end
        if type == 3
            tabu(n + new_SAR(1)) = 1;  %Set new SAR to tabu
            tabu_time(n + new_SAR(1)) = 1; %Set tabu time to 1
        end
        for l = 1:w
             tabu(n+m+l) = weapons(l);  %Set new weapons set to tabu
        end
        tabu(n+m+w+1)= no_uav; %Set number of UAVs to tabu
    end
    for i = 1:(n+m)
        if tabu_time(i) == 3  %Tabu tenure
            tabu(i) = 0;
        end
    end
end
%End tabu search
```

A.2 Initial Solution

```
%create an initial feasible solution

function [sensors, tabu, budget, weight, size, SAR, weapons, no_uav,
orig_budget, tabu_time] = initial_parameters(X, Y, budget, weight,
size, sensors, SAR, weapons, tabu, tabu_time, n, m, w, no_sensors,
no_SAR, weapon_weight, total_budget)

SAR_tabu = zeros(1, m);
sens_tabu = zeros(1,n);
weapon_tabu = zeros(1, w);
orig_budget = budget;

%EO/IR sensors
new_sens = unidrnd(n,1,n);  %Generate a size n array with a list of
                             sensors
for i = 1:n
        if X(new_sens(i),1) <= budget  %Sensor meets budget constraint
            if X(new_sens(i),2) <= weight  %Sensor meets weight
                                            constraint
                if X(new_sens(i),3) <= size  %Sensor meets size
                                              constraint
                    sensors(new_sens(i)) = 1;  %Add new sensor to
                                                  sensor array
                    budget = budget - X(new_sens(i),1);  %Update budget
                    weight = weight - X(new_sens(i),2);  %Update weight
                    size = size - X(new_sens(i),3);  %Update size
                    sens_tabu(new_sens(i)) = 1;  %Update sensor tabu
                                                    list
                    no_sensors = no_sensors +1;  %Update number of
                                                    sensors
                end
            end
        end
        if no_sensors == 3  %Three sensors have been added to the UAV
            break
        end
end

%SAR
new_SAR = unidrnd(m,1,m);  %Generate a size m array with a list of SAR
```

```
for j = 1:m
        if Y(new_SAR(j), 1) <= budget  %SAR meets budget constraint
            if Y(new_SAR(j), 2) <= weight  %SAR meets weight constraint
                if Y(new_SAR(j), 3) <= size  %SAR meets size constraint
                    SAR(new_SAR(j)) = 1;  %Add new SAR to SAR array
                    budget = budget - Y(new_SAR(j), 1);  %Update budget
                    weight = weight - Y(new_SAR(j), 2);  %Update weight
                    size = size - Y(new_SAR(j), 3);  %Update size
                    SAR_tabu(new_SAR(j)) = 1;  %Update SAR tabu list
                    no_SAR = no_SAR +1;  %Update number of SAR
                end
            end
        end
        if no_SAR == 1  %One SAR has been added to the UAV
            break
        end
end

%Weapons
weapon_500 = floor((weapon_weight*rand())/500);  %Determine number of
                                                  500 lb weapons
weapon_weight = weapon_weight - 500*weapon_500;  %Update weapon weight
weapon_250 = floor((weapon_weight*rand())/250);  %Determine number of
                                                  250 lb weapons
weapon_weight = weapon_weight - 250*weapon_250;  %Update weapon weight
weapon_60 = floor(weapon_weight/60);  %Determine number 60 lb weapons
weapons = [weapon_500, weapon_250, weapon_60];
weapon_tabu = weapons;

%Number of UAVs
max_no_uav = floor(total_budget/(orig_budget-budget));
no_uav = unidrnd(max_no_uav,1,1);

%end initial feasible solution
tabu = [sens_tabu(1,:), SAR_tabu(1,:), weapon_tabu(1,:), no_uav];
%Update global tabu list

for i = 1:(n+m) %Update tabu time
    if tabu(i) == 1
        tabu_time(i) = 1;
    end
end

return
%End initial solution
```

A.3 Diversification

```
%Start multi-swap diversification
function [sensors, budget, weight, size, new_sens, r, SAR, new_SAR,
weapons, no_uav] = multi_swap(X, Y, sensors, SAR, budget, weight, size,
n, m, w, weapons, weapon_weight, no_uav, orig_budget, total_budget)
```

```
%Perform a diversifying swap
orig_weapon_weight = weapon_weight;
SAR_record = zeros(1,1);
record = zeros(1,3);
j = 1;
s = 1;

different = 0;
%change sensors
for i = 1:n
    if sensors(i) == 0
    else
        sensors(i) = 0;  %Remove sensor from list
        budget = budget + X(i, 1);  %Update budget
        weight = weight + X(i, 2);  %Update weight
        size = size + X(i, 3);  %Update size
        record(j) = i;
        j = j + 1;
    end
    if  j == 4  %Three sensors have been removed
        break
    end
end
while different == 0
    r = unidrnd(3,1,1);  %Choose the number of sensors to include
    new_sens = unidrnd(n,1,r);  %Generate r new sensors
    different = 1;

    for k = 1:j-1
        for l = 1:r
            if record(k) == new_sens(l)  %The new sensors are the same
                                          as those removed
                different = 0;
                break
            end
        end
    end

    if r == 3
        if (new_sens(1) == new_sens(3)) || (new_sens(2) == new_sens(3))
            different = 0;
        else if new_sens(1) == new_sens(2)
                different = 0;
            end
        end
    else if r == 2
        if new_sens(1) == new_sens(2)
            different = 0;
        end
        end
    end

    if different == 0
    else
        for i = 1:r
            if X(new_sens(i), 1) <= budget  %New sensor combination
                                             meets the budget
                                             constraint
                if X(new_sens(i), 2) <= weight  %New sensor combo meets
                                                 the weight constraint
                    if X(new_sens(i), 3) <= size  %New sensor combo
                                                   meets the size
                                                   constraint
                        different = 1;
                        sensors(new_sens(i)) = 1;  %Update the sensor
                                                    array
                        budget = budget - X(new_sens(i),1);  %Update
                                                              budget
                        weight = weight - X(new_sens(i),2);  %Update
                                                              weight
                        size = size - X(new_sens(i),3);  %Update size
                    end
                end
            end
        end
    end
end
%end change sensors
%change SAR
different = 0;
for i = 1:m
    if SAR(i) == 0
    else
        SAR(i) = 0;  %Remove current SAR from list
        budget = budget + Y(i, 1);  %Update budget
        weight = weight + Y(i, 2);  %Update weight
        size = size + Y(i, 3);  %Update size
        SAR_record(s) = i;  %Record the value of the SAR removed
        s = s + 1;
    end
    if  s == 2  %The one SAR component has been removed
        break
    end
end

while different == 0

    new_SAR = unidrnd(m,1,1);  %Generate a new SAR value
    different = 1;

    if SAR_record(1) == new_SAR(1)  %SAR value generated is the same as
                                     the previous SAR value
        different = 0;
    end

    if different == 0
    else  %SAR value generated is different from the previous SAR Value
```

```
        different = 0;
        if Y(new_SAR(1), 1) <= budget  %New SAR value meets the budget
                                       constraint
            if Y(new_SAR(1), 2) <= weight  %New SAR value meets the
                                           weight constraint
                if Y(new_SAR(1), 3) <= size  %New SAR value meets the
                                              size constraint
                    different = 1;
                    SAR(new_SAR(1)) = 1;  %Update SAR
                    budget = budget - Y(new_SAR(1),1);  %Update budget
                    weight = weight - Y(new_SAR(1),2);  %Update weight
                    size = size - Y(new_SAR(1),3);   %Update size
                end
            end
        end
    end
end
%end change SAR
%Change weapons
different = 0;
while different == 0
    different = 1;
    weapon_500 = floor((weapon_weight*rand())/500);  %Determine number
                                                      of 500 lb weapons
    weapon_weight = weapon_weight - 500*weapon_500;  %Update weapon
                                                      weight
    weapon_250 = floor((weapon_weight*rand())/250);  %Determine number
                                                      of 250 lb weapons
    weapon_weight = weapon_weight - 250*weapon_250;  %Update weapon
                                                      weight
    weapon_60 = floor(weapon_weight/60);  %Determine number 60 lb
                                           weapons
    new_weapons = [weapon_500, weapon_250, weapon_60];
    if new_weapons == weapons
        weapon_weight = orig_weapon_weight;
        different = 0;
    end
    if different == 1
        weapons = new_weapons;
    end
end
%end change weapons
%Change no_uav
no_old_uav = no_uav;
max_no_uav = floor(total_budget/(orig_budget-budget));
different = 0;
while different == 0
    no_uav = unidrnd(max_no_uav,1,1);
    different = 1;
    if no_uav == no_old_uav
        different = 0;
    end
end
%end change no_uav
```

```
Return
%End multi-swap diversification
```

A.4 Intensification

```
%Start single-swap intensification
function [sensors, budget, weight, size, new_sens, r, no_uav] =
single_swap(X, sensors, budget, weight, size, n, orig_budget,
total_budget, no_uav)

record = zeros(1,3);
j = 1;
different = 0;
r = 1;  %Only one EO/IR sensor will change
for i = 1:n
    if sensors(i) == 0
    else
        record(j) = i;  %Records the values of the sensors that are in
                        use
        j = j + 1;
    end
    if  j == 4  %Break if three sensors are in use
        break
    end
end

sen_delete = unidrnd(j-1,1,1);  %Record the value of the sensor that
                                 will be removed
sensors(record(sen_delete)) = 0;  %Delete the sensor that is removed
budget = budget + X(record(sen_delete),1);  %Update budget
weight = weight + X(record(sen_delete),2);  %Update weight
size = size + X(record(sen_delete),3);  %Update size

while different == 0
    new_sens = unidrnd(n,1,1);  %Choose a new sensor to add to the set
                                 of sensors
    different = 1;

    for k = 1:(j-1)
        if record(k) == new_sens  %The new sensor is the same as a
                                   current or deleted sensor
            different = 0;
            break
        end
    end
    if different == 0
    else  %The new sensor is different than the sensor deleted
        different = 0;
        if X(new_sens, 1) <= budget  %The new sensor meets the budget
                                      constraint
            if X(new_sens, 2) <= weight  %The new sensor meets the
                                          weight constraint
                if X(new_sens, 3) <= size  %The new sensor meets the
                                            size constraint
```

```
                    different = 1;
                    sensors(new_sens) = 1;  %The sensor array is
                                             updated
                    budget = budget - X(new_sens,1);  %Update budget
                    weight = weight - X(new_sens,2);  %Update weight
                    size = size - X(new_sens,3);   %Update size
                end
            end
        end
    end
end

%Evaluate no_uav
max_no_uav = floor(total_budget/(orig_budget-budget));
if no_uav > max_no_uav
    no_uav = unidrnd(max_no_uav,1,1);
end

return
%Stop single-swap intensification
```

Bibliography

Aarts, E. H. L., Korst, J. H. M., and P.J.M. Laarhoven. Simulated Annealing. E. Aarts, J.K. Lenstra, eds. *Local Search in Combinatorial Optimization*. Princeton, NJ: Princeton University Press, 2003.

Anandalingam, G. Simulated Annealing. S. I. Gass, C. M. Harris, eds. *Encyclopedia of Operations Research and Management Science*, 2nd ed. Boston, MA: Kluwer Academic, 2001.

Andradóttir, Sigrún. Simulation Optimization. Chapter 9 in Jerry Banks, ed. *Handbook of Simulation: Principles, Methodology, Advances, Applications, and Practice*. New York: John Wiley and Sons, Inc., 1998.

April, Jay, Fred Glover, James P. Kelly, and Manuel Laguna. "Practical Introduction to Simulation Optimization," *Proceedings of the Winter Simulation Conference*. 71-78, 2003.

Banks, Jerry, John S. Carson II, Barry L. Nelson, and David M. Nicol. Comparison and Evaluation of Alternative System Designs. *Discrete-Event System Simulation* (4th Edition). New Jersey: Pearson Prentice Hall, 2005.

Banks, Jerry. Principles of Simulation. Chapter 1 in Jerry Banks, ed. *Handbook of Simulation: Principles, Methodology, Advances, Applications, and Practice*. New York: John Wiley and Sons, Inc., 1998.

Boesel, Justin, Barry L. Nelson, and Seong-Hee Kim. "Using Ranking and Selection to 'Clean Up' After Simulation Optimization," *Operations Research*, 51.5: 814-835 (Sept/Oct 2003).

Chaput, Armand J. Course slides, Lockheed Martin UAV Design Course. Lockheed Martin Aeronautics Company, Fort Worth TX, 2002.

Eglese, R. W. 1990. Simulated Annealing: A Tool for Operational Research. *European Journal of Operational Research*, 46: 271-281.

Fu, Michael C., Fred W. Glover, and Jay April. "Simulation Optimization: A Review, New Developments, and Applications," *Proceedings of the Winter Simulation Conference*. 83-95, 2005.

Fu, Michael C. "Optimization for Simulation: Theory vs. Practice," *INFORMS Journal on Computing*, 14.3: 192-215 (Summer 2002).

Fu, Michael C. "Simulation Optimization," *Proceeding of the Winter Simulation Conference*. 53-61, 2001.

Glover, Fred, James P. Kelly, and Manuel Laguna. "New Advances for Wedding Optimization and Simulation," *Proceedings of the Winter Simulation Conference*. 255-260, 1999.

Glover, Fred and Manuel Laguna. *Tabu Search*. Boston: Kluwer Academic Publishers, 1997.

Glover, Fred. Tabu search. S. I. Gass, C. M. Harris, eds. *Encyclopedia of Operations Research and Management Science*, 2nd ed. Boston, MA: Kluwer Academic, 2001.

Goldsman, David and Barry L. Nelson. Comparing Systems via Simulation. Chapter 8 in Jerry Banks, ed. *Handbook of Simulation: Principles, Methodology, Advances, Applications, and Practice*. New York: John Wiley and Sons, Inc., 1998.

Greenberg, Harvey J. "Computational Testing: Why, How, and How Much," *ORSA Journal on Computing*, 2:1: 94-97 (Winter 1990).

Hertz, A., Taillard, E., and D. de Werra. Tabu search. E. Aarts, J.K. Lenstra, eds. *Local Search in Combinatorial Optimization*. Princeton, NJ: Princeton University Press, 2003.

Hooker, J. N. "Testing Heuristics: We Have it All Wrong," *Journal of Heuristics*, 1:1: 33-42, (Fall 1995).

Kleijen, Jack P. Experimental Design for Sensitivity Analysis, Optimization, and Validation of Simulation Models. Chapter 6 in Jerry Banks, ed. *Handbook of Simulation: Principles, Methodology, Advances, Applications, and Practice*. New York: John Wiley and Sons, Inc., 1998.

Law, Averill M. *Simulation Modeling and Analysis* (4th Edition). New York: McGraw-Hill, 2007.

Mileshosky, Katelyn M. "What is Synthetic Aperture Radar?" Sandia National Laboratories. n. pag., 2005. http://www.sandia.gov/radar/whatis.html. 16 July 2007.

Morris, David, Air Vehicles Directorate, Air Force Research Laboratory. "A Revolutionary Hunter-Killer UAS for Enhanced Survivability, Persistence, and Effectiveness." Air Force Research Laboratory. Wright Patterson AFB, OH. 12 April 2006.

Mühlenbein, Heinz. Genetic Algorithms. E. Aarts, J.K. Lenstra, eds. *Local Search in Combinatorial Optimization*. Princeton, NJ: Princeton University Press, 2003.

Sait, S.M. and Youssef, H. Genetic Algorithms (GA). *Iterative Computer Algorithms with Applications in Engineering*. Los Alamitos, CA: IEEE Computer Society, 1999a.

Sait, S.M. and Youssef, H. Introduction. *Iterative Computer Algorithms with Applications in Engineering*. Los Alamitos, CA: IEEE Computer Society, 1999b.

Sait, S.M. and Youssef, H. Simulated Annealing (SA). *Iterative Computer Algorithms with Applications in Engineering*. Los Alamitos, CA: IEEE Computer Society, 1999c.

Sait, S.M. and Youssef, H. Tabu search (TS). *Iterative Computer Algorithms with Applications in Engineering*. Los Alamitos, CA: IEEE Computer Society, 1999d.

"Unmanned Aircraft Systems Roadmap, 2005-2030." Office of the Secretary of Defense. 213 pag., 2005. http://www.acq.osd.mil/usd/Roadmap%20Final2.pdf. 31 October 2007.

REPORT DOCUMENTATION PAGE			*Form Approved* *OMB No. 074-0188*
The public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of the collection of information, including suggestions for reducing this burden to Department of Defense, Washington Headquarters Services, Directorate for Information Operations and Reports (0704-0188), 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to an penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number. **PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.**			
1. REPORT DATE *(DD-MM-YYYY)* 03-10-2008	**2. REPORT TYPE** **Master's Thesis**		**3. DATES COVERED** *(From – To)* Oct 2006 - Mar 2008
4. TITLE AND SUBTITLE A SIMULATION OPTIMIZATION APPROACH TO THE DESIGN OF UNMANNED AIRCRAFT SYSTEMS			**5a. CONTRACT NUMBER**
			5b. GRANT NUMBER
			5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S) **7.** Evans, Emily C.			**5d. PROJECT NUMBER**
			5e. TASK NUMBER
			5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAMES(S) AND ADDRESS(S) Air Force Institute of Technology Graduate School of Engineering and Management (AFIT/EN) 2950 Hobson Street, Building 642 WPAFB OH 45433-7765			**8. PERFORMING ORGANIZATION REPORT NUMBER** AFIT/GOR/ENS/08-22
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) AFRL/RD Attn: Mr. James Zeh (937) 904-6556 e-mail: james.zeh@wpafb.af.mil			**10. SPONSOR/MONITOR'S ACRONYM(S)**
			11. SPONSOR/MONITOR'S REPORT NUMBER(S)
12. DISTRIBUTION/AVAILABILITY STATEMENT APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.			
13. SUPPLEMENTARY NOTES			

14. ABSTRACT

Military strategy and operations have evolved significantly over the past decade. This evolution has led to a change in the military resources required to carry out missions successfully. In line with these requirements, demand has increased for unmanned aerial vehicles (UAV) with enhanced capability to perform surveillance and to strike targets of interest. This research effort aids in the design of a next generation UAV by employing a simulation optimization approach. The goal of this research is to maximize the number of targets destroyed in a conflict scenario by a newly designed UAV that is subject to size, weight, and budget constraints. The solution approach involves the development of a simulation model representing a conflict scenario, which includes various types and quantities of targets, and weather conditions. The model is used to test the effectiveness of various UAV configurations in detecting and destroying targets. A tabu search meta-heuristic is constructed to optimize the configuration of the UAV, in terms of the number and type of sensors, synthetic aperture radar, and weapons.

15. SUBJECT TERMS

Simulation Optimization, Simulation, Metaheuristics, Tabu Search, Design of Experiments

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON** Shane N. Hall, Maj, USAF (ENS)
a. REPORT	**b. ABSTRACT**	**c. THIS PAGE**			**19b. TELEPHONE NUMBER** *(Include area code)* (937) 785-3636; e-mail: Shane.Hall@afit.edu
U	U	U	UU	83	